Analysis of Crewed Missions Enabled by a Bimodal Nuclear Thermal Electric Propulsion System

Undergraduate Honors Thesis

Presented in Partial Fulfillment of the Requirements for
Graduation with Distinction in the
Department of Aerospace Engineering at
The Ohio State University

Authored by:

Justin Clark

May 2019

Advisor: Dr. John Horack

Abstract

From bone decay, to radiation exposure and muscle atrophy, long duration spaceflight poses significant health risks to humans. Technologies and practices have been developed that limit the impacts of some of these health issues, but perhaps the simplest method would be one of avoidance. Reducing mission durations through nuclear propulsion technologies has been researched since the 1950s, when NASA's NERVA program tested various nuclear thermal engines for crewed applications. The technology showed significant promise but fell by the wayside due to political and environmental reasons. However, interest in nuclear thermal propulsion (NTP) has been renewed as of late, with NASA's Marshall Space Flight Center and private companies such as BWX Technologies developing a new cryogenic NTP system for use with crewed Mars missions, possibly reducing one-way trip times from 6 months to 4 months. And while this would be a significant improvement, it's possible that a standalone NTP system doesn't meet the full potential of a nuclear fission powered spacecraft. Instead, a combined nuclear thermal and nuclear electric propulsion (NEP) system, also known as a bimodal nuclear propulsion system, could prove to be more efficient, and offer shorter mission times than just a single NTP system. A spacecraft with a bimodal nuclear propulsion system would have a single nuclear reactor and the capability to switch between both modes, altering the heat output of the reactor based on the required propulsion system. The NTP system would offer high thrust and an ISP in the 900s, and would be useful for escaping the sphere of influence of a planet. The NEP system, on the other hand, would offer very low thrust, but would provide long periods of propulsion in interplanetary space, as well as an ISP in the 5000s.

In this paper, mission opportunities gained by a combined NTP/NEP system will be explored. Specifically, trajectories of crewed missions to orbit Mars and Jupiter using such a

system will be analyzed, with the intent to minimize overall mission time. Fuel and payload weight calculations will also be performed to ensure that the delta-v budgets for these missions are adhered to. Some focus on the design of the reactor/propulsion system will also be given, and the overall performance and benefit relative to chemical and standalone NTP powered craft will be presented.

Acknowledgements

Over the course of this endeavor, many individuals helped in one way or another, and without their assistance, this project would not have been possible. First, I'd like to extend much love and many thanks to my wonderful family for all their support, not just for the duration of this undergraduate thesis, but also for the entirety of my undergraduate career here at Ohio State. There have been many strenuous moments over the past four years, and it's always been reassuring to know that they're there for me at all times. So, to mom, dad, Logan, Clayton, Lynn, Patrick, and Mee-Maw, thank you.

Next, I'd like to thank the Battelle Center for Science, Engineering, and Public Policy for sponsoring this research and offering guidance throughout this process. To Dr. Elizabeth Newton, thank you for your constant support, and I'm ever so grateful for our insightful conversations about leadership and the human machine. To Jenny Shields, thank you for not just your logistical support with events and conferences, but your moral support to keep pushing through this research even when it seemed too difficult. To Dr. John Horack, thank you for giving me the courage to pursue my interests when I wasn't confident in my ability to innovate. The wisdom and knowledge you've imparted on me concerning the technicalities and human patterns in astronautical engineering have inspired and emboldened me to continue down this career path.

Finally, to everyone, thank you for your constant encouragement and kind words. They say you become like those you're around, and if that's the case, then the person I'm growing into is going to be insightful, kind, supportive, and knowledgeable, and I only have you all to thank for that.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Deep space presents an extremely hostile environment to the human body, with some hazards having long-lasting effects, or even being lethal. Specifically, astronauts on long duration deep space missions are exposed to a significant increase in solar and galactic radiation upon leaving the Earth's magnetosphere, damaging an astronaut's nervous system and increasing their likelihood of developing cancers. Low gravity is also an issue, and NASA claims that the average astronaut will lose 1% of their bone density per month as compared with elderly persons' 1.5% per year bone density loss. Additionally, the build up of bacteria and severe isolation and confinement can lead to an increase in both physical and mental illnesses, respectively (Abadie et al., 2018).

Space agencies today spend vast amounts of resources on countering these issues, but perhaps the best strategy would be one of avoiding them altogether by shortening the duration of missions altogether. This is where new advanced propulsion systems come into play and is why NASA and other space agencies invest both time and money in researching into new propulsion systems. To demonstrate the significance in developing newer, more efficient propulsion systems, consider Equation 1, the Tsiolkovsky Rocket Equation, below, with ΔV being the change in velocity of a spacecraft necessary to travel to a destination, Isp being the ratio of thrust to weight of fuel consumed, g0 being Earth's gravitational acceleration (9.81 m/s^2), m0 being the initial mass of the craft before a burn, and mf being the final mass of the craft after a burn.

$$\Delta V = Isp * g_0 * \ln\left(\frac{m_0}{m_f}\right) \quad (1)$$

The Tsiolkovsky Rocket Equation is significant as, for a given engine, it describes how much fuel is required in order to gain a desired delta-V. Of course, in order to alter orbits and ultimately travel between moons, planets, and stars, there is always a certain delta-V required to do so, and therefore delta-V can be described as the "cost" required of any space mission. It's also crucial the one notices the exponential relationship between mass and delta-V, and as delta-V requirements increase, the mass of fuel needed to meet those requirements increases exponentially and is one of the main aspects of astrodynamics that makes space travel as challenging as it is. Rightfully, this property has been dubbed "the tyranny of the rocket equation" and provides insight into why engine efficiency is so important. Isp ties directly in to this exponential relationship, so an increase in Isp leads to an exponential decrease in the mass of fuel required. However, it's also important to note that generally for spacecraft engines, an increase in Isp correlates with a decrease in thrust, so to gain a certain delta-v a high thrust engine doesn't need to operate for as long. For leaving spheres of influences of planets, low thrust systems generally take longer than high thrust systems for the reason previously provided. Therefore, for interplanetary mission durations, high thrust systems can provide shorter mission durations than low thrust systems for the same delta-V.

Now, consider the example of a crewed mission to Mars utilizing different propulsion systems. One potential system could be a traditional chemical propulsion system, offering high thrust but low Isp. Another potential system could be solar electric propulsion (SEP), that offers high Isp but not a significant amount of either power or thrust. Nuclear electric propulsion (NEP) could also be used, offering high Isp, high power, but low thrust. Finally, nuclear thermal propulsion (NTP) could be utilized, offering high thrust and greater Isp than a chemical system, but still less than either of the electric propulsion systems. Figure 1 presents a comparison of delta-V requirements with trip durations for each of these propulsion systems, demonstrating the utility of each system. Figure 1 originates from (Houts et al., 2018).

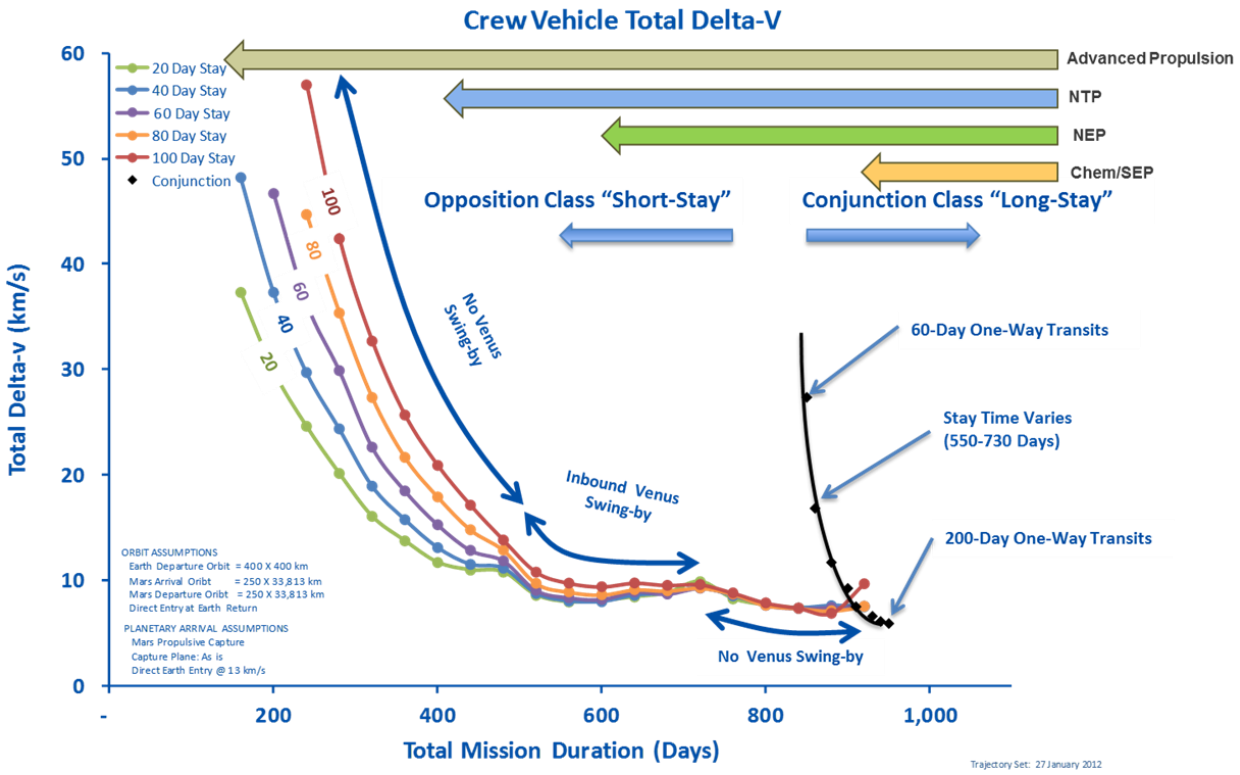

Figure 1: Average Delta-V Requirements as Functions of Mars Stay Time (Houts et al., 2018)

Figure 1 demonstrates the tradeoff between engine thrust and efficiency, as one can see that the NEP system offers shorter mission durations than a chemical or SEP system, and this is

likely due to the fact that the high power NEP system can offer more delta-V for the same initial mass, and even though the chemical system can depart planets sooner, the NEP's additional delta-V more than makes up for the chemical system's advantage in this area. It's also noteworthy that the NTP system offers even shorter mission durations than the NEP system, as the quick planetary departure times and greater Isp (than chemical systems) both benefit the reduction of mission duration. However, how exactly do NTP and NEP systems operate? What makes them such promising technologies?

Nuclear Thermal Propulsion, or NTP, operates by utilizing nuclear fission to heat and thereby accelerate fuel. Nuclear Thermal Propulsion was originally conceptualized back in the 1960s and 1970s under the Rover and NERVA (Nuclear Engine for Rocket Vehicle Application) programs and was a joint venture between the Atomic Energy Commission and NASA. The programs were tasked with equipping vehicles with nuclear rocket engines to demonstrate their effectiveness and practicality in space vehicle applications. However, the programs were shut down in 1973 before a prototype test could be completed, but not before final designs and terrestrial tests were conducted. One such design was the NERVA Flight Engine, which was able to generate a thrust of 75,000 lbf at an Isp 825 seconds and a chamber pressure of 450 psi. This design operated by having a nuclear reactor core of low enriched uranium fissile to heat liquid hydrogen fuel, causing the fuel to expand into a superheated gas and is then accelerated through a nozzle. Additionally, the reactor could regulate the rate of the fission reaction by utilizing a system of control drums. Specifically, these drums functioned by absorbing neutrons from the core, and thereby interrupting the fission process. Figure 2 below is a schematic of such an engine similar to the NERVA Flight Engine (Gerrish, 2014).

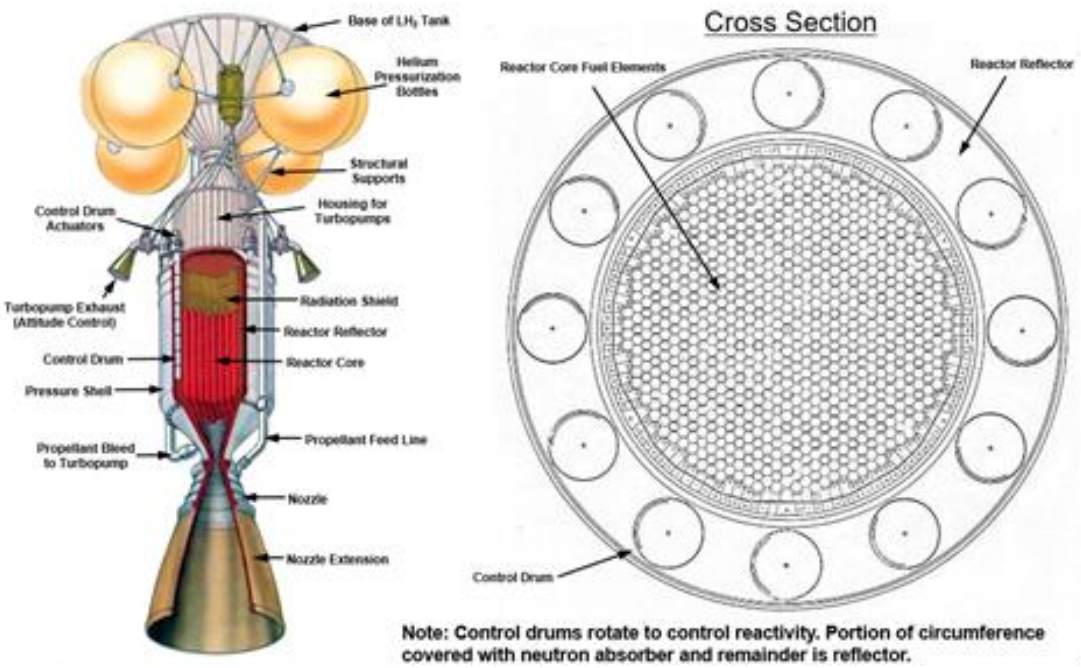

Figure 2: Bimodal Nuclear Thermal Reactor and Propulsion Schematics (Houts et al, 2018)

Figure 2 details the layout of the NERVA Flight Engine and shows a cross section of the uranium reactor core and the control drum feature. Specifically, these control drums are made from a neutron reflector material, that increases the neutron concentration in the core by sending escaping neutrons back. On the other hand, when these control drums are rotated to the neutron absorption coat side, the number of neutrons in the core will decrease, ultimately slowing the fission reaction and cooling the reactor.

Nuclear electric propulsion systems, or NEP, are another promising space propulsion technology that space agencies are developing and operate by utilizing a nuclear fission reactor to generate electricity for high power electric propulsion systems. One such system is the VAriable Specific Impulse Magnetoplama Rocket, or VASIMR, engine. This high-power electric propulsion system is under development by Ad Astra Rocket Company and operates by

accelerating plasma particles by utilizing a strong magnetic field. Figure 3 below depicts how this system operates (Castro Nieto et al., 2013).

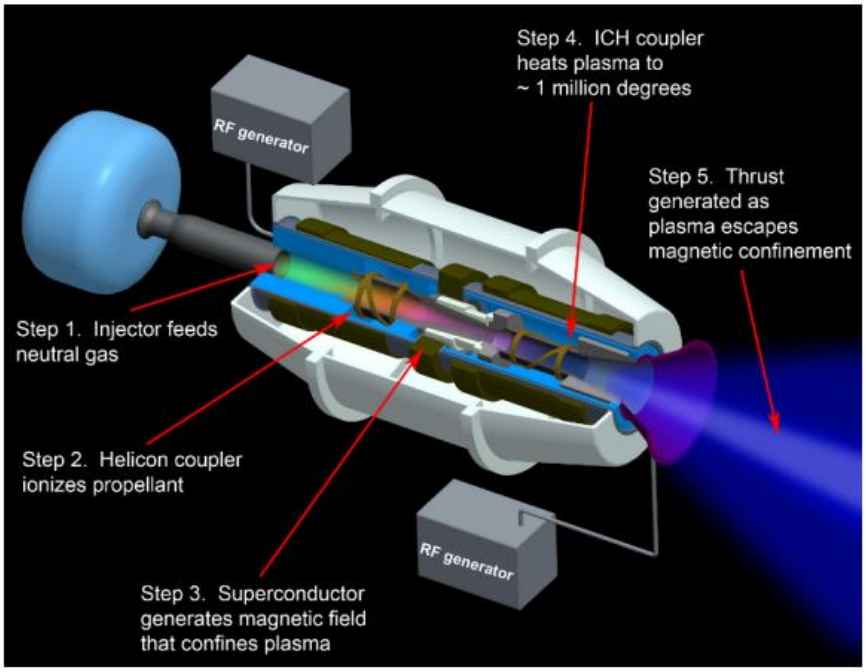

Figure 3: Components and Schematic of the VASIMR Engine (Castro Nieto et al., 2013)

Figure 3 specifically shows how this process works. First, an injector pumps a neutral gas into the engine, where a helicon coupler then ionizes the propellant to be able to move under the influence of a magnetic field. Now, a superconductor is charged, generating a strong magnetic field that moves the charged plasma. The plasma is then heated and then accelerated out the back of the engine at a high speed to produce thrust. For the VASIMR VX-200 model, a thrust of 6N is generated at an Isp of 5000s, and the system requires 200 kW of power (Castro Nieto et al., 2013).

Both NTP and NEP benefit from high-power density that nuclear fission offers spacecraft, and when saving mass is crucial, as evidenced by the Tsiolkovsky Rocket Equation, utilizing nuclear power propulsion systems become even more advantageous.

Chapter 1.1: Applications and Significance

This paper hypothesizes that mission durations can be further reduced by utilizing a bimodal nuclear thermal electric propulsion system, or BNTEP system, taking the advantages of both NTP and NEP systems, and putting them into one hybrid propulsion system. A spacecraft with BNTEP could minimize mission durations by utilizing high thrust NTP in the spheres of influences of planets to depart quickly, and by then using high Isp NEP in interplanetary space to generate more delta-V for less fuel mass. During NTP burn phases, the nuclear reactor would run at high energy for short amounts of time, generating hundreds of MW of heat energy. On the other hand, during periods of low thrust NEP burns, the reactor's control drums can be turned to limit the decrease the amount of power to the scale of hundreds of kW (Borowski, 2003).

The idea of a BNTEP system is not new, and one of the first major researchers/proponents of a bimodal nuclear thermal reactor is Dr. Stanley Borowski, who proposed a similar system in his 2003 presentation titled, *"Bimodal" Nuclear Thermal Rocket (BNTR) Propulsion for Future Human Mars Exploration Missions.* In this presentation Borowski theorizes that the power/propulsion system would operate as follows: a turbopump pumps hydrogen from the liquid hydrogen (LH2) tank to the reactor; the reactor heats and accelerates the hydrogen atoms as they pass by; finally, the hydrogen atoms proceed out the bottom of the reactor and into a nozzle, further accelerating the particles. The reactor is cooled during this process by having the LH2 flow around it before being pumped through the reactor. A visualization of this engine can be found in Figure 4 on the following page and originates from Borowski's presentation.

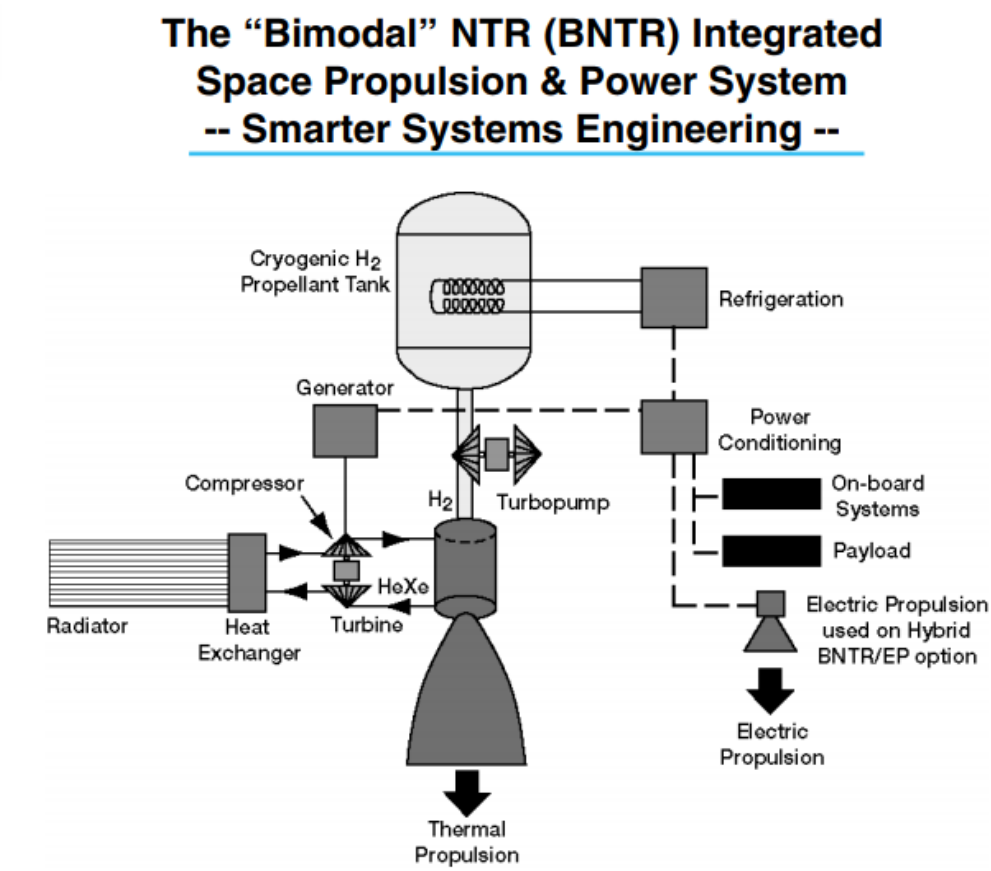

Figure 4: Configuration of a Bimodal Nuclear Thermal Electric Propulsion System (Borowski, 2003)

Borowski's presentation then concludes with an example mission using a BNTEP system to Mars and can be seen in Figure 5 on the following page. This mission lasts 902 days and consists of a 210-day outbound trajectory from Earth to Mars, a 502 day stay time, and then a 190 day return trajectory. Such a trajectory saves propellant mass that would be necessary for a traditional chemical mission, but also exposes astronauts to over 2.5 years' worth of radiation, low gravity, and other harmful effects.

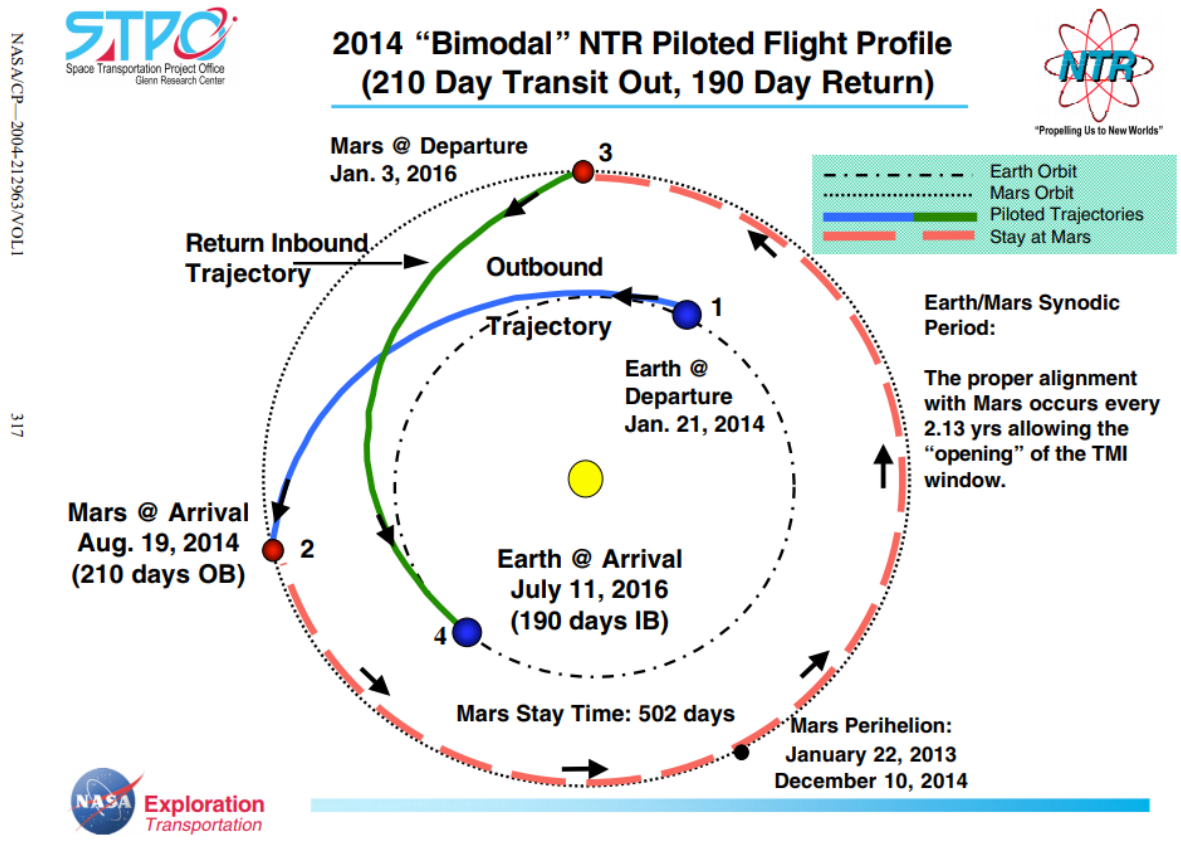

Figure 5: Example Mars Mission Utilizing BNTEP Technology (Borowski, 2003)

This research differentiates itself from Borowski's work on the subject by attempting to shorten mission durations even further than what resulted from Borowski's work. Borowski's Mars mission utilizes a conjunction class trajectory, which essentially minimizes delta-V requirements by having a crew on the surface of Mars for over a year. On the other hand, this research will attempt to keep astronauts at Mars for a much shorter duration, about 20-40 days, before returning to Earth. Such a mission is termed an opposition class mission, and requires greater amounts of delta-V, but greatly reduces mission durations, and therefore, exposure times. Additionally, Borowski's research was conducted long before the recent innovations and improvements in propulsion technologies were known, so promising technologies, such as the VASIMR engine or efficiency improvements to NTP systems are not considered.

Chapter 1.2: Overview of Thesis

This paper will first begin by detailing the methodology of this research in Chapter 2: Approach and Methodology, where initial orbital parameters, payloads, spacecraft system capabilities and other assumptions will be stated and justified with known data from space agencies and contracted companies. Next, the results of this research will be presented in Chapter 3: Results and Comparison, focusing on trajectories, craft/fuel masses, and mission durations. Additionally, a craft with a BNTEP system will be compared against a craft utilizing a standalone NTP system and one utilizing a chemical propulsion system. In each case, trajectories in each "patched conic" will be presented, consisting of each craft's trajectories as they depart Earth, orbit the sun, encounter/depart Mars, and return to Earth. Following this section, Chapter 4: Significance and Discussion will explore the ramifications of this research, and ultimately delve into how this may affect astronaut health on long duration deep space missions. Chapter 5 will present the conclusions of this research, its limitations, and possible improvements for future studies. Finally, Appendix A: MATLAB Script will provide the MATLAB script and function files utilized in this research to enable reproduction and verification of the yielded results.

Chapter 2: Methodology

This research was conducted in three stages, dividing up the analysis of different propulsion types. The initial propulsion type that was investigated was the BNTEP system and mission architecture. This was then succeeded by the standalone NTP system and mission architecture and finalized with the chemical system and mission architecture. Each mission focused on attaining realistic initial masses, while minimizing the total mission duration. This was accomplished by calculating the necessary delta-v values for each leg of the journey, then using a final, desired spacecraft mass to calculate the initial Earth departure mass.

Chapter 2.1: Designing BNTEP Mission Architecture

When creating the mission architecture for the BNTEP system, the mission was broken down into four separate components: Earth departure, heliocentric Earth departure/ Mars arrival, Mars arrival/departure, and heliocentric Mars departure/ Earth arrival. Selections of the Earth departure and Mars arrival/departure orbits were made based on mission analyses from (Houts et al., 2018) in Figure 1. In accordance with this work, a 400 x 400 km Earth departure orbit was chosen, and a 250 x 33813 km Mars arrival/departure orbits were chosen. For the Earth departure portion of the simulation, a desired hyperbolic excess velocity was chosen, the corresponding delta-v was calculated and the trajectory (in Earth's frame) was calculated from the craft's NTP system.

At this point, the heliocentric Mars-bound portion of the mission began, and the NEP system was utilized, yielding a constant thrust until the craft arrived at Mars' sphere of influence. However, throughout the duration of this stage, the flight path angle of the NEP system could be manipulated, and the problem of low-thrust trajectory optimization arose. To tackle this complication in the trajectory, David Eagles' Two-dimensional, Low-thrust Earth-to-Mars Trajectory Analysis with MATLAB program, utilizing the Sparse Non-linear OPTimizer, or SNOPT program implemented into the MATLAB script (Eagle, 2016) (Gill et al., 2005).

This program, given initial/final boundary conditions, a control variable, system dynamics, and an objective function, could find a solution using direct collocation and transcription. In this case, the boundary conditions were initial/final positions and initial/final radial and transverse velocities at Earth and Mars, respectively. Additionally, the control variable used was the flight path angle, and the objective function was the maximization of radial distance from the sun, with the goal of reaching Mars as quickly as possible while arriving at a certain velocity.

The system dynamics consist of differential equations 2 through 4 below. r is the radial distance of the spacecraft from the sun, u is the radial velocity of the spacecraft, v is the tangential velocity of the spacecraft, a(t) is the acceleration of the craft at a given moment in time, and $\boldsymbol{\psi}$ is the firing angle of the spacecraft's thrust. As for the differential equations, Equation 2 represents rate of change in the radial distance of the spacecraft from the sun, Equation 3 represents the radial acceleration of the spacecraft, and Equation 4 represents the tangential acceleration of the spacecraft.

$$\dot{\boldsymbol{r}} = \boldsymbol{u} \tag{2}$$

$$\dot{\boldsymbol{u}} = \frac{\boldsymbol{v^2} - \frac{\boldsymbol{1}}{\boldsymbol{r}}}{\boldsymbol{r}} + \boldsymbol{a(t)} * \mathbf{sin}(\boldsymbol{\psi}) \tag{3}$$

$$\dot{\boldsymbol{v}} = \frac{-\boldsymbol{v} * \boldsymbol{u}}{\boldsymbol{r}} + \boldsymbol{a(t)} * \mathbf{cos}(\boldsymbol{\psi}) \tag{4}$$

With this optimization complete, Mars intercept and orbital insertion could begin. The design of the entry hyperbola stemmed from the need for a gravity assist from Mars, with the goal to reduce the heliocentric transverse velocity after Mars departure. So, given a velocity of the craft arriving at Mars, a hyperbolic trajectory in Mars' orbital plane was engineered, and NTP burns were placed at the hyperbola's periapsis to place the craft in the desired 250 x 33813 km orbit. The craft would then remain in this orbit for however long was desired, and finally burn once more at periapsis to depart Mars on the same hyperbolic trajectory which it arrived in.

For the final segment of the journey, the craft would once again utilize its constant thrust NEP system, starting from the new heliocentric orbit created by the departure burn at Mars. However, the same approach used earlier to arrive at Mars couldn't be used, as this was now a rendezvous problem.

In the previous heliocentric trajectory, the spacecraft had the option to leave Earth whenever Mars was aligned properly, but if the craft is to return in a minimal amount of time, it cannot wait until the Earth is in a better position. As such, an opposition-type mission was selected, and a Lambert's Problem trajectory must be used upon return. To solve this rendezvous problem, Matthew Kelly's OptimTraj program was used to optimize the trajectory of the craft given boundary conditions, a set amount of time, and a control variable. For this instance, the boundary conditions are the positions of Mars and Earth at departure and arrival, respectively. Additionally, given a specific inbound flight time of the craft, Earth's final position could be calculated. The initial and final velocities of the craft were also boundary conditions, with the initial velocity being the velocity departing Mars. The craft was also specified from Figure 1 to enter Earth's atmosphere at 13 km/s or less, so this was used as the final velocity limit of the spacecraft. Matthew Kelly's program then solves for an optimal path using a variety of numerical methods, and finally plots the trajectory of the craft (Kelly, 2015).

The difference between the OptimTraj program and the SNOPT program is that the SNOPT program is more robust and accurate in its calculation for an optimal path but is more computationally expensive. On the other hand, OptimTraj is less accurate at solving for the best path but is easier to compute, and in the case where the program has an additional differential equation, this is necessary. Specifically, the rendezvous/return trajectory program utilizes the same differential equations (Equations 2-4) as in the outbound optimization but must also consider angular position relative to Earth and Mars. Equation 6 represents the differential equation for the angular position.

$$\dot{\boldsymbol{\theta}} = \boldsymbol{v}/\boldsymbol{r} \tag{6}$$

Chapter 2.2: Designing NTP Mission Architecture

When designing the mission architecture of the standalone NTP craft to compare with that of the BNTEP craft, two missions were designed. One of which begins with the same payload and fuel masses as the BNTEP craft, while removing the masses associated with the NEP system. The second mission also removes the masses of the NEP system but calculated the necessary IMLEO mass of the craft to achieve the same mission duration as the BNTEP mission. Both missions were calculated by repeating the process used with the BNTEP mission, but just allowing the craft to reach Mars by coasting after Earth departure and by using David Eagle's Lambert's Problem program to calculate the delta-v required to reach a certain set of orbital parameters in a given amount of time (Eagle, 2014).

Chapter 2.3: Designing Chemical Propulsion Mission Architecture

The chemical system was calculated in much the same way as the NTP system, but with an Isp of 450 for chemical propulsion instead of 950 for NTP. Mission durations and initial IMLEO masses were also calculated in the same manner as the two missions designed for the standalone NTP system.

Chapter 2.4: Payload and Initial IMLEO Mass Consideration

Selection of habitat payload mass was made based on Figure 6 below and originates from (Arney et al., 2017), which defines the habitat payload as a function of mission duration. From this figure, it was selected that for a trip of about 350-400 days, a payload of 30,000 kg should be used account for habitat mass, crew, consumables, and spare parts. For trips of 400-450 and 850-900 days, the habitat masses should be 32,500 kg and 40,000 kg, respectively. Additionally, when considering the masses of the NEP system and NTP system, there were six VASIMR VX-200 engines and three NERVA inspired NTP engines used aboard the BNTEP spacecraft. The NEP engines each have a mass of 620 kg, and the NTP engines each have a mass of 2000 kg. For chemically propelled craft, the RS-25 engine has a mass of 3500 kg, and three of these engines were used. Finally, a re-entry craft mass of 10,000 kg was added to each mission's dry mass.

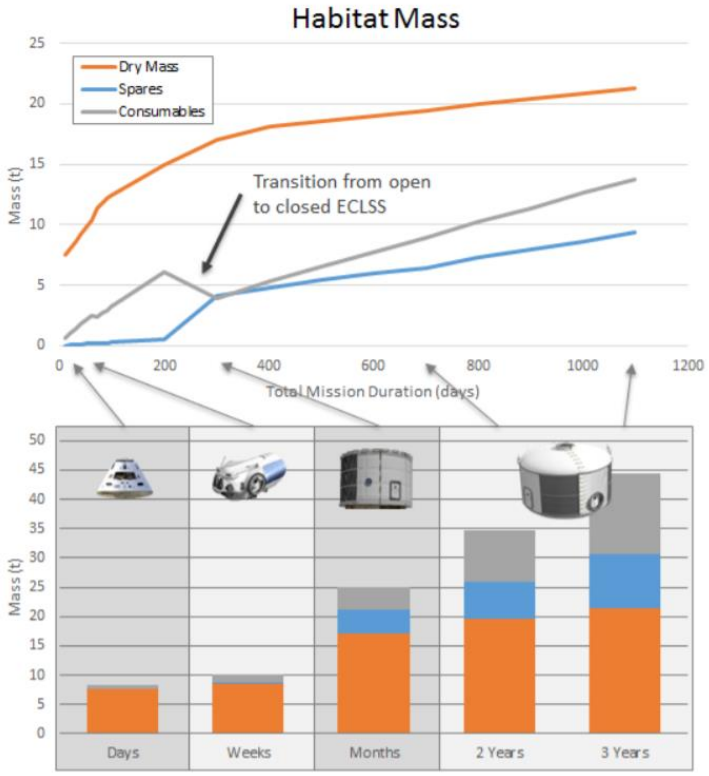

Figure 6: Crewed Habitat Mass as a Function of Mission Duration (Arney et al., 2017)

Now, adding in a 25% margin of error leads to the final, dry craft mass of 62,150 kg for a 350-400 day BNTEP mission, 57,500 kg for a 350-400 day NTP mission, 60,625 kg for a 400-450 day NTP mission, 63,125 kg for a 350-400 day chemical propulsion mission, and 75,625 kg for an 850-900 day chemical propulsion mission.

Chapter 3: Results

The next three subsections will present the results of the methodology detailed in the previous chapter, with each section presenting the results employing a different propulsion system. Additionally, the results will be condensed and displayed in Table 1: Mission Parameters as Functions of Mission Propulsion Systems at the end of the section.

Chapter 3.1: BNTEP System Results

Based on the approach discussed in section 4.1, the results in Table 1 below and Figures 7 - 11 below are presented. Utilizing the BNTEP system, it was found that a spacecraft carrying an 62,150 kg payload could make a 20-day Mars orbital mission in 365.32 days for a delta-v of 21.276 km/s and yielding an initial departure mass of approximately 300,000 kg.

Specifically, the BNTEP spacecraft begins its journey by firing its NTP system in LEO for around 48 minutes to gain a delta-v of 3.891 km/s. This then allows the craft to depart Earth's sphere of influence and enter interplanetary space. Next, the craft fires its NEP engines for 142.32 days, gaining a delta-v of 1.905 km/s. Once the craft intercepts Mars, it waits until periapsis is reached, then fires its NTP engines for 32 minutes and decelerates by 4.13 km/s to enter Martian orbit. After 20 days, the NTP engines accelerate the craft by 4.13 km/s, burning for 20 minutes at Martian periapsis, and the craft completes its leading side pass of Mars. Finally, the NEP engines fire for 203 days, providing a delta-v of 7.222 km/s and allowing the craft to intercept Earth at about 9 to 10 km/s, safe enough for re-entry.

Figures 9 and 11 show the trajectory parameters of the spacecraft during the constant thrust components of its mission, and it can be seen how the flight path angle can be altered to affect the other trajectory characteristics.

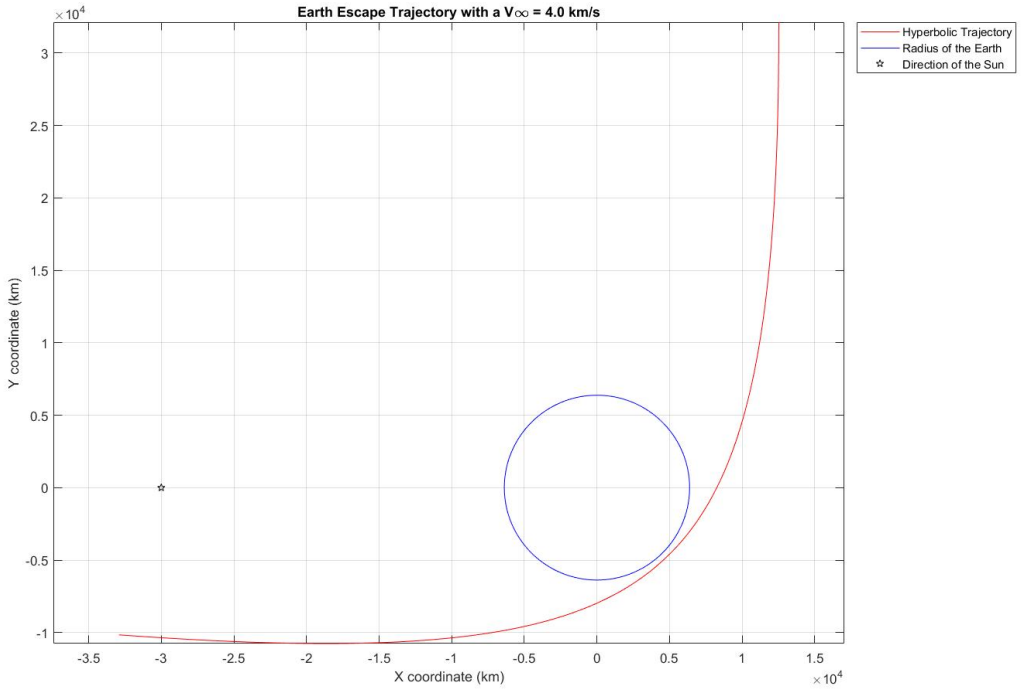

Figure 7 Earth Departure Trajectory of a BNTEP Craft

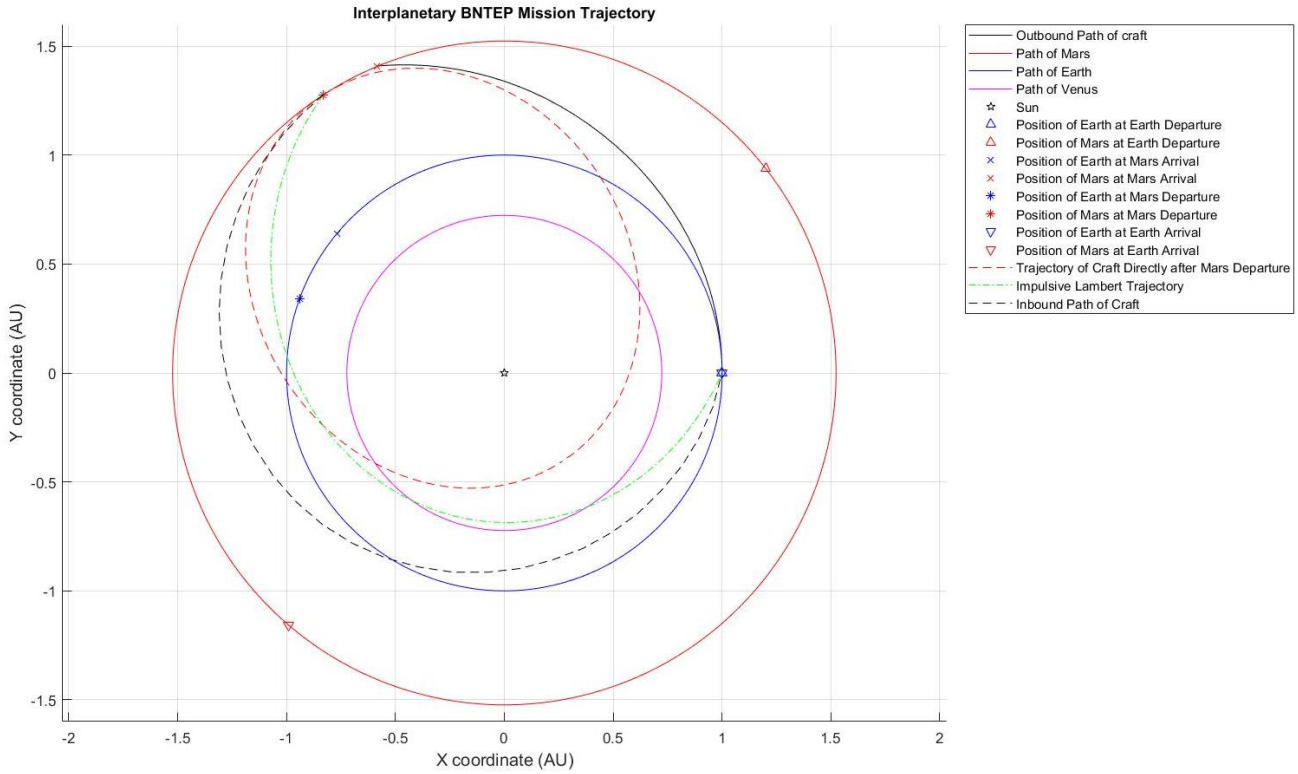

Figure 8: Heliocentric Trajectory of a BNTEP Craft

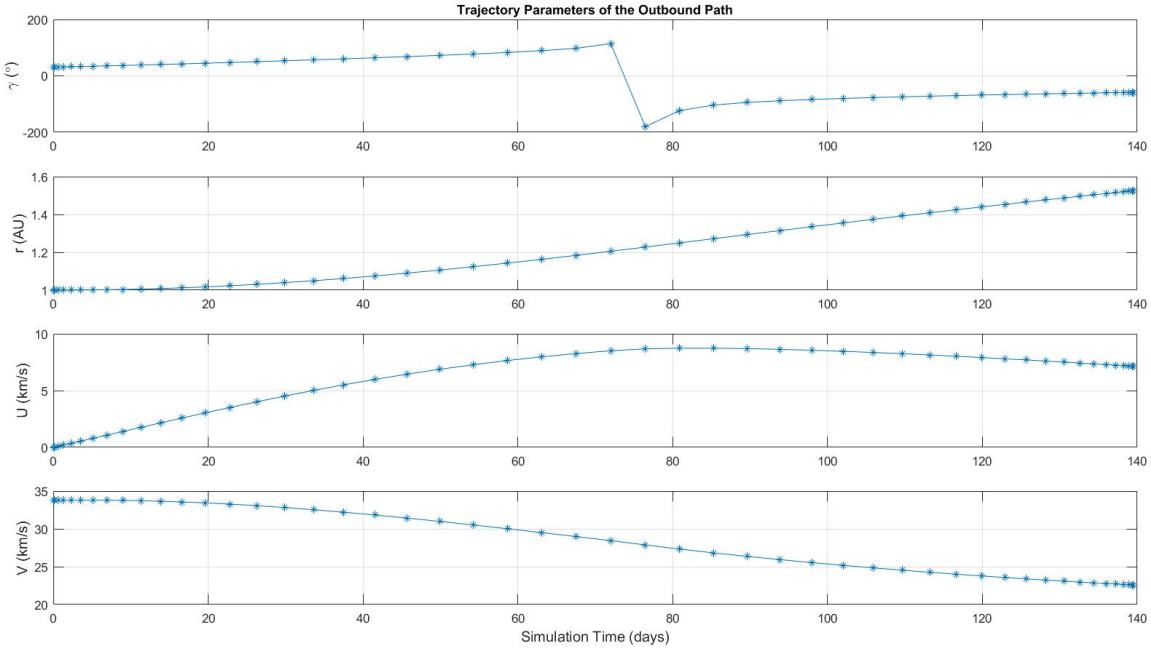

Figure 9: Mars Bound Trajectory Parameters and Flight Path Angle Optimization

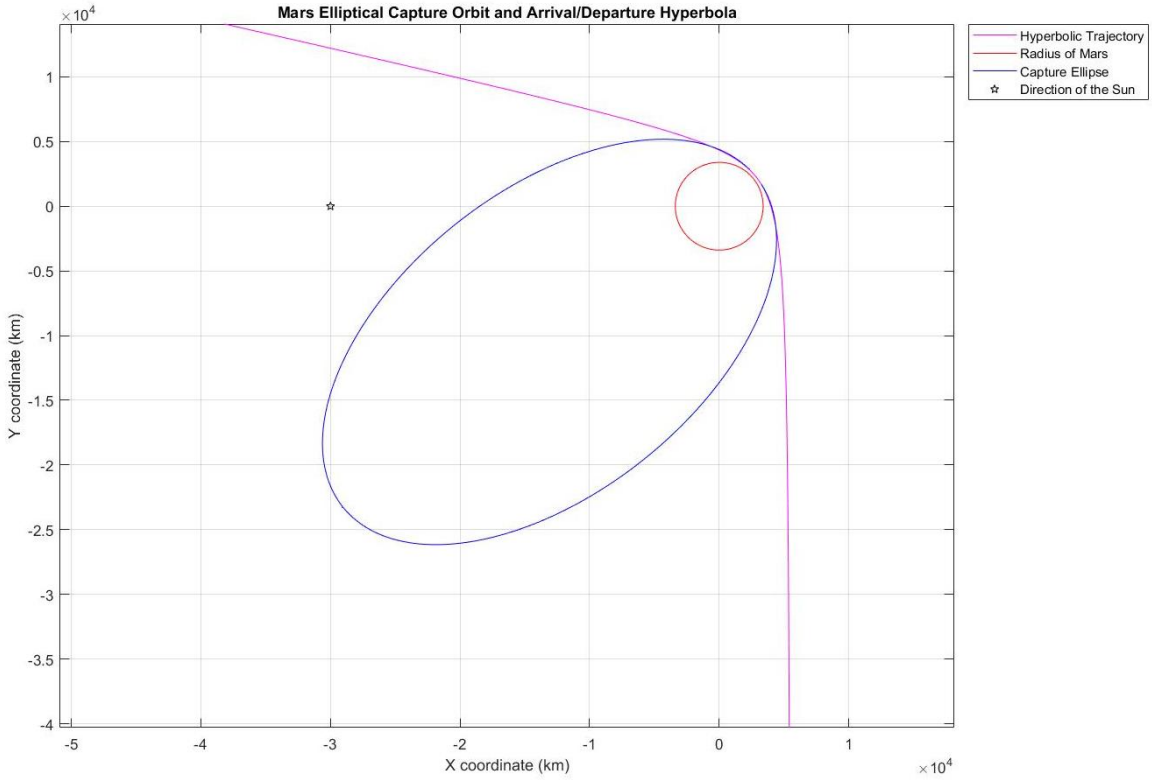

Figure 10: BNTEP Mars Arrival and Departure Orbit and Hyperbola

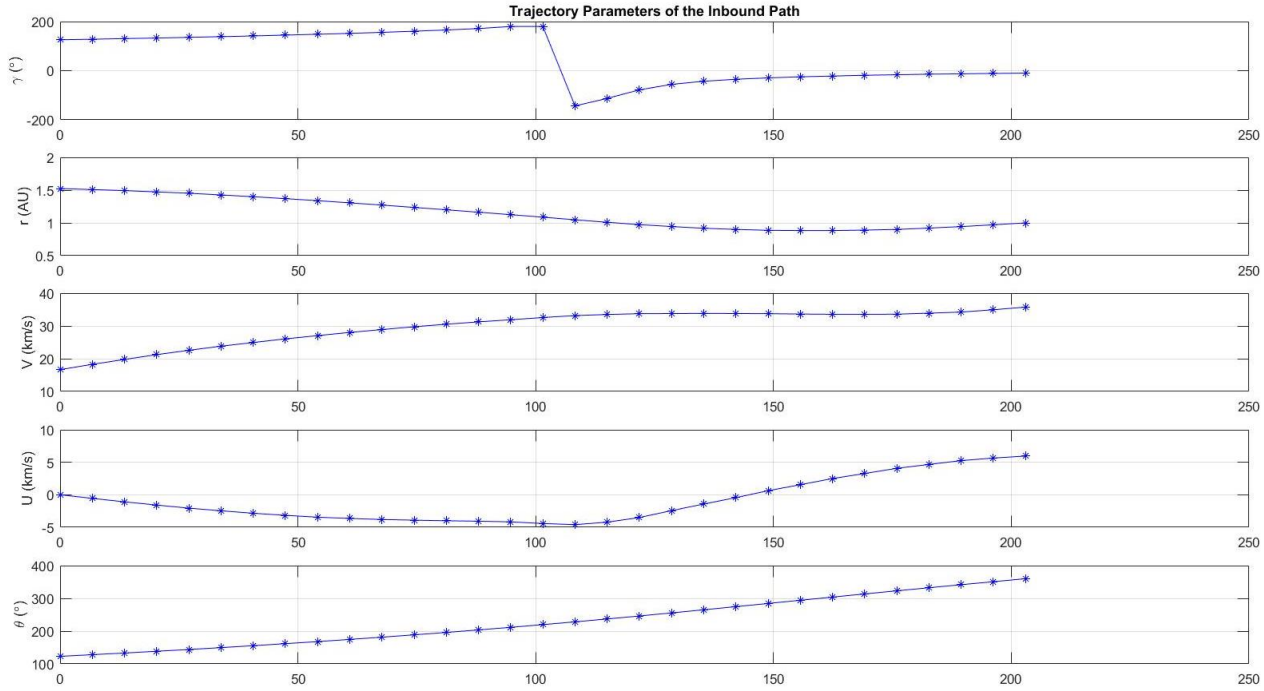

Figure 11: Return Trajectory and Flight Path Angle Optimization

Chapter 3.2: Standalone NTP System Results

To accomplish the same mission in the same amount of time as the BNTEP mission, it was found that a spacecraft using exclusively an NTP system would need an IMLEO mass of 764,617 kg to expend a delta-v of 24.115 km/s in 364.86 days, traveling towards Mars for 154.86 days, staying for 20 days, and returning to Earth for 190 days. In LEO, the engines fire for 2.05 hrs to provide a delta-v of 3.891 km/s to propel the craft towards Mars. Once at Mars periapsis, a delta-v of 5.107 km/s is obtained by burning for 1.67 hrs, and after 20 days, accelerates by 5.107 km/s by burning for 0.96 hrs. Finally, the engines fire again for 0.87 hrs to apply a final delta-v of 10.011 km/s. Figures 12 through 14 represent this mission and are presented below.

However, if the craft were to have an initial mass of approximately 300,000 kg, it could complete the mission in 430.54 days using a delta-v of 14.975 km/s, traveling towards Mars for 160.54 days, staying for 20 days, and taking 250 days to return to Earth. Specifically, the spacecraft would initially burn for 48 minutes in LEO to gain a delta-v of 3.840 km/s. As the craft intercepts Mars and reaches periapsis, the engines burn for 37 minutes to decelerate by 4.605 km/s. After 20 days of waiting in Martian orbit, the craft burns for 22 minutes to accelerate by 4.605 km/s. Finally, the craft burns again for 7 minutes once arriving in interplanetary space to gain a delta-v of 1.925 km/s. The craft would the travel from Mars to Earth and would pass within the orbit of Venus. Figures 15 through 17 below present this mission's trajectory.

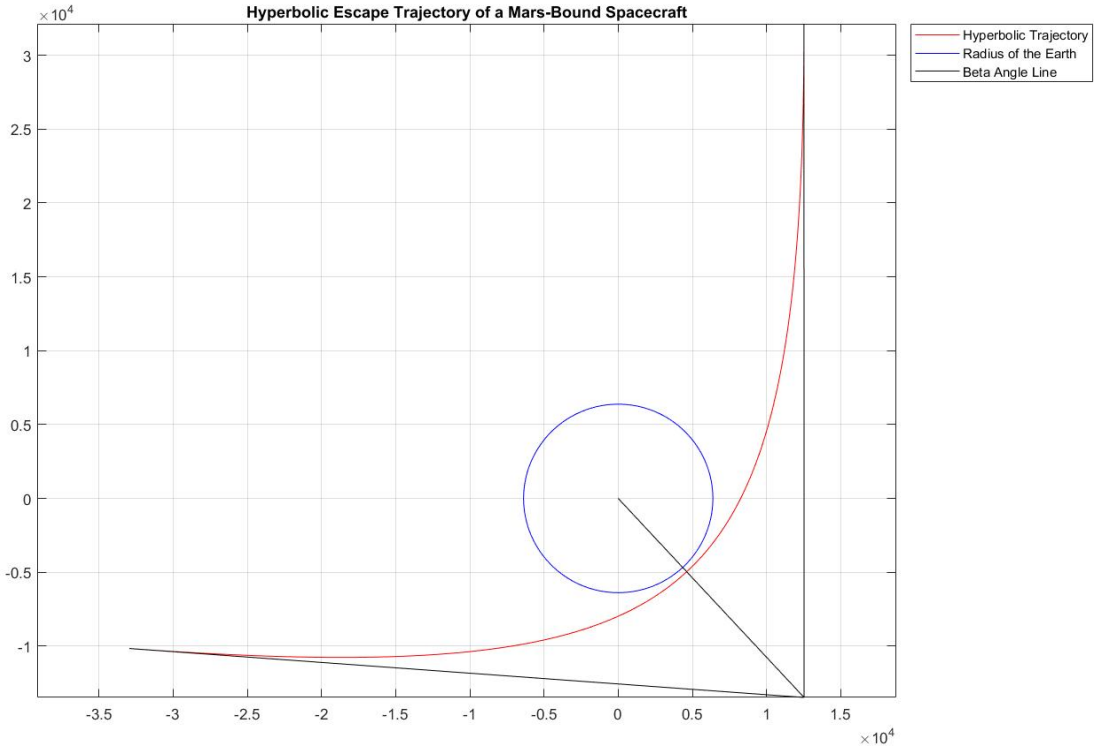

Figure 12: Short Duration NTP Earth Departure Trajectory

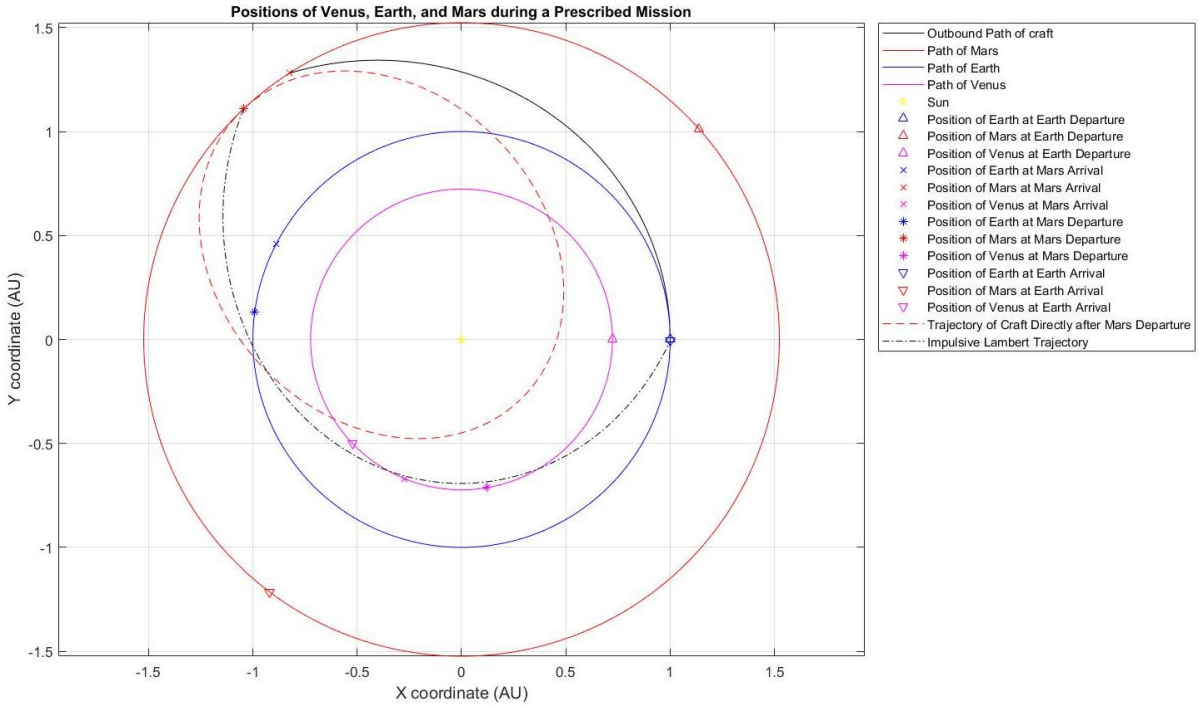

Figure 13: Short Duration NTP Heliocentric Trajectory

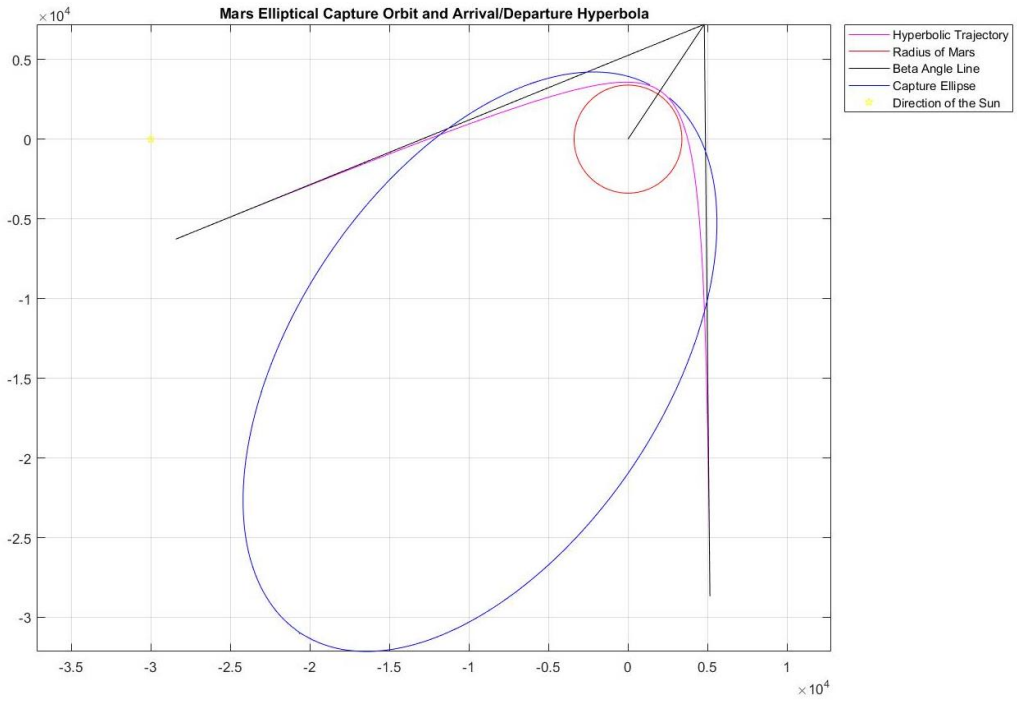

Figure 14: Short Duration NTP Mars Arrival and Departure Trajectories

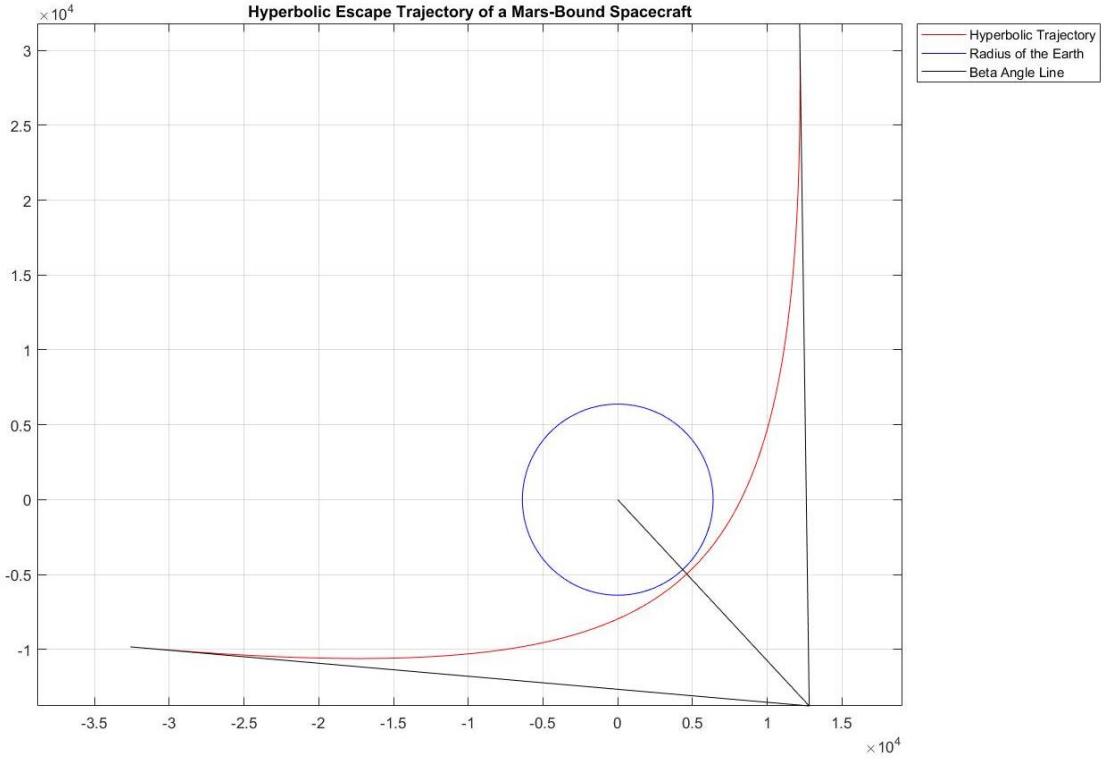

Figure 15: Long Duration NTP Earth Departure Trajectory

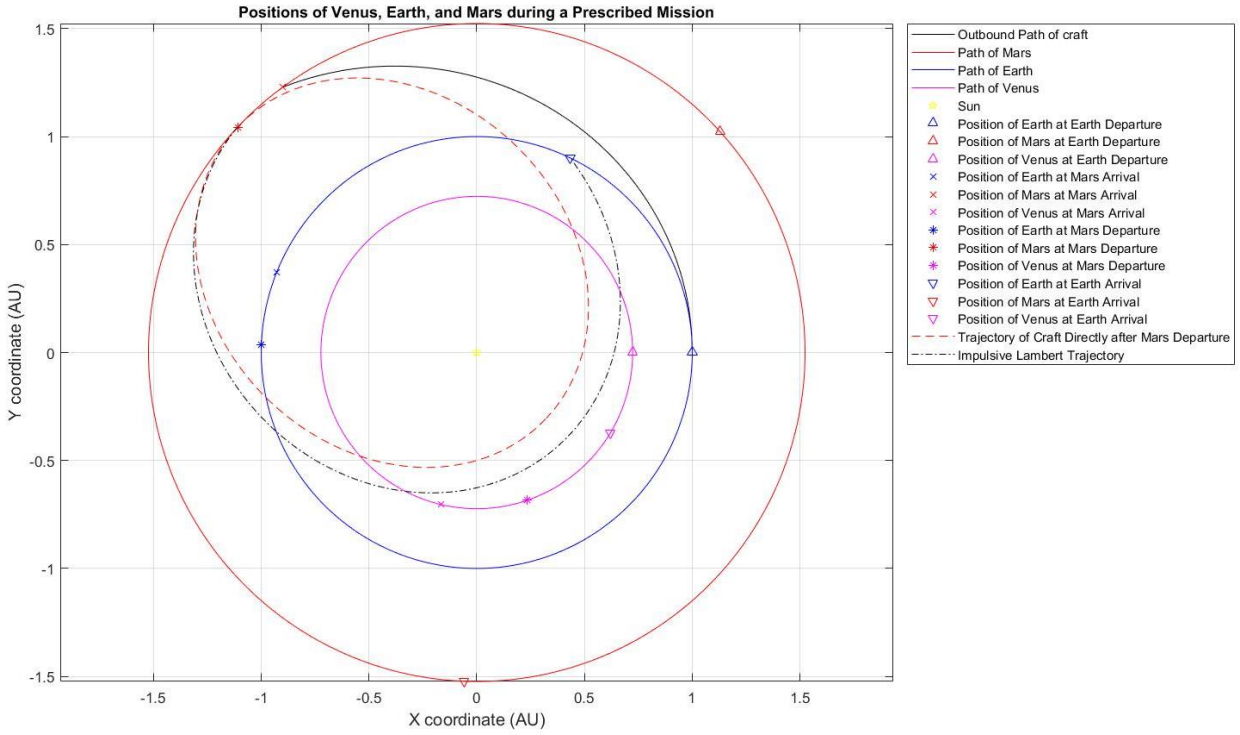

Figure 16: Long Duration NTP Heliocentric Trajectory

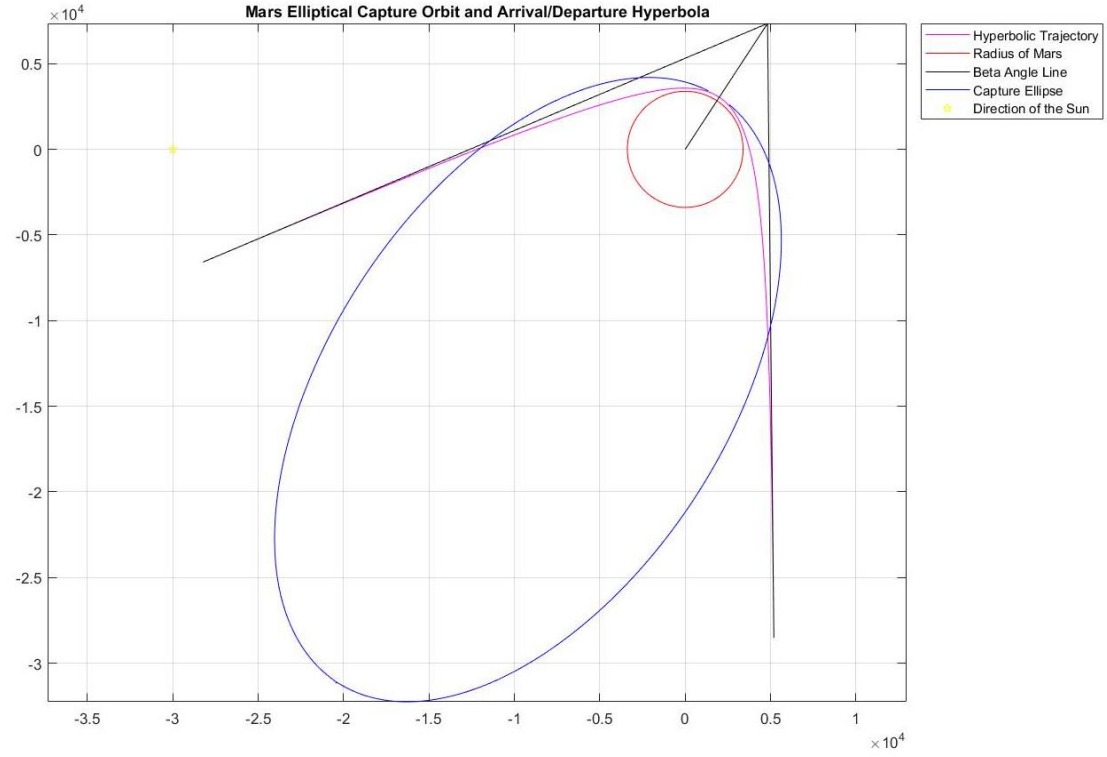

Figure 17: Long Duration NTP Mars Arrival and Departure Trajectories

Chapter 3.3: Chemical Propulsion System Results

The same delta-v and mission duration as the NTP mission were used in calculating the initial mass of the spacecraft using a chemical propulsion system. For a chemical propelled spacecraft to expend 24.115 km/s worth of delta-v in 364.86 days, it requires an initial mass of approximately 14,880,544 kg. Specifically, the spacecraft would initially burn for 94 minutes in LEO to gain a delta-v of 3.891 km/s. As the craft intercepts Mars and reaches periapsis, the engines burn for 45 minutes to decelerate by 5.107 km/s. After 20 days of waiting in Martian orbit, the craft burns for 14 minutes to accelerate by 5.107 km/s. Finally, the craft burns again for 6 minutes once arriving in interplanetary space to gain a delta-v of 10.011 km/s. Figures 18 through 20 represent the corresponding trajectory.

Finally, if a chemical propelled craft were to have an initial mass of around 300,000 kg, it was determined the mission could be completed in 891.71 days using 12.863 km/s worth of delta-v, and by having a Martian stay time of 500 days instead of 20 days. Specifically, the spacecraft would initially burn for 2.2 minutes in LEO to gain a delta-v of 3.706 km/s. After 181 days, the craft intercepts Mars and reaches periapsis, the engines burn for 1.3 minutes to decelerate by 3.0525 km/s. After 500 days of waiting in Martian orbit, the craft burns for 0.92 minutes to accelerate by 3.0525 km/s. Finally, the craft burns again for 0.67 minutes once arriving in interplanetary space to gain a delta-v of 3.052 km/s. The craft then coasts for 210 days before arriving at Earth. Figures 21 through 23 model this trajectory.

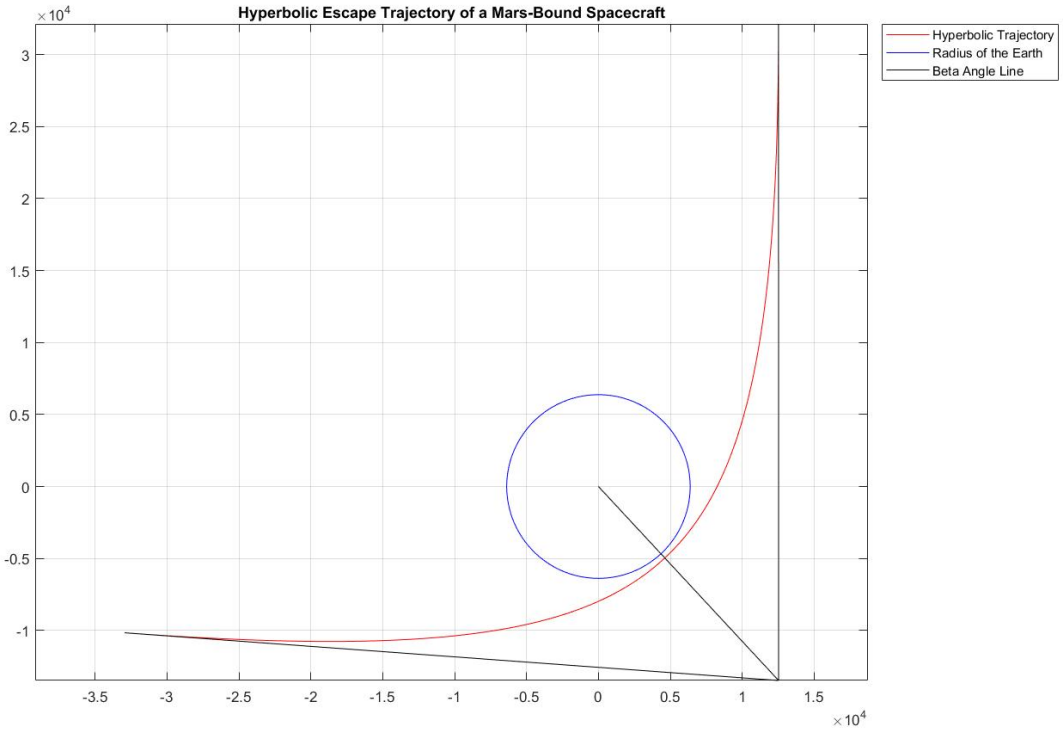

Figure 18: Short Duration Chemical Earth Departure Trajectory

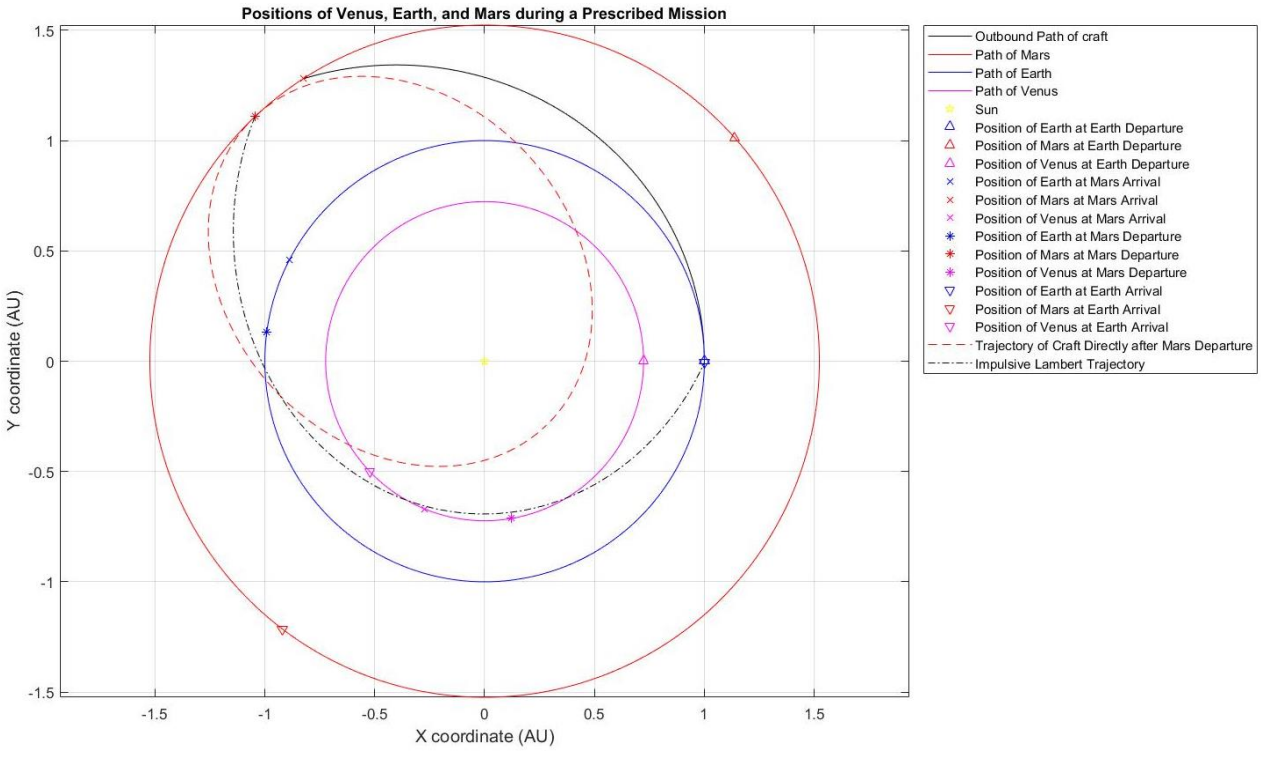

Figure 19: Short Duration Chemical Heliocentric Trajectory

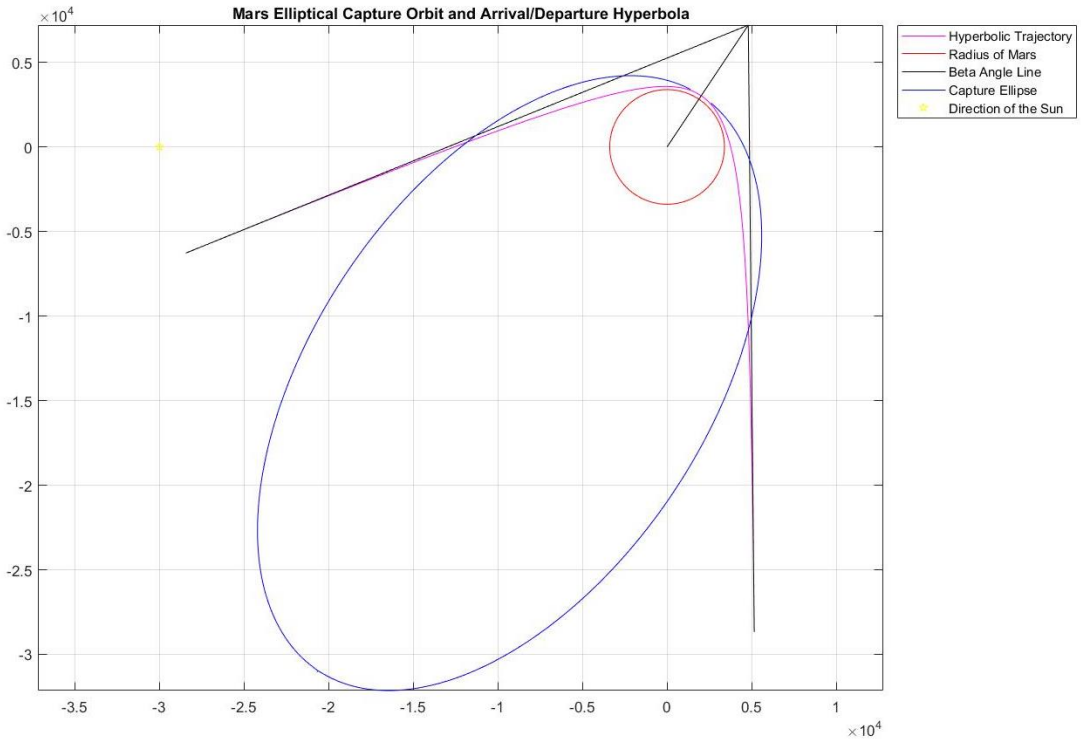

Figure 20: Short Duration Chemical Mars Arrival and Departure Trajectories

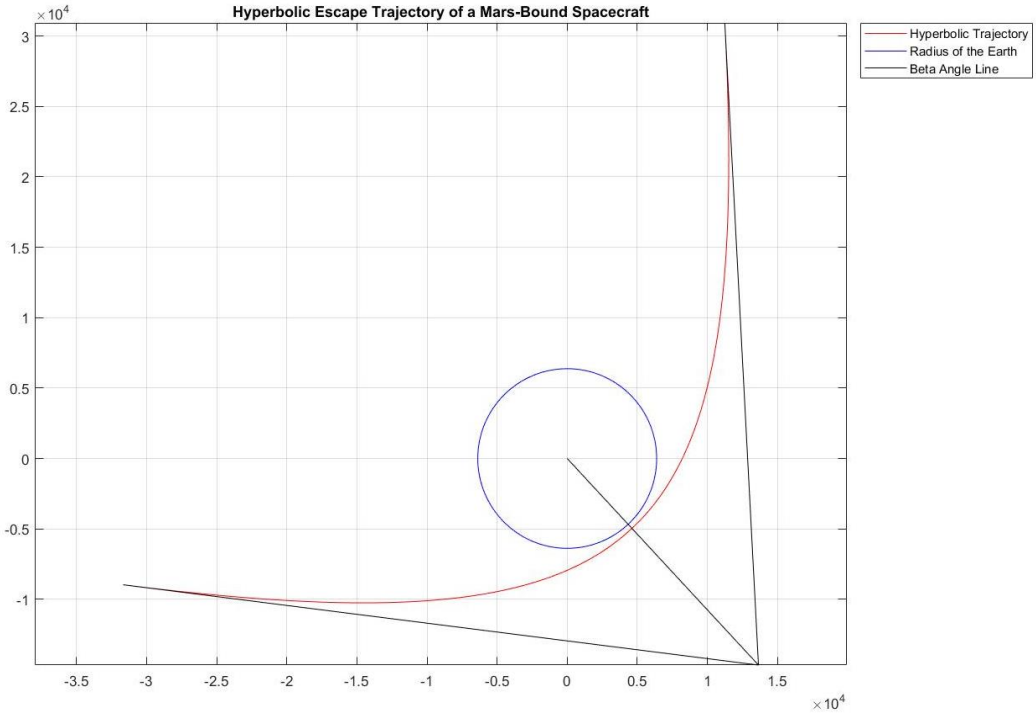

Figure 21: Long Duration Chemical Earth Departure Trajectory

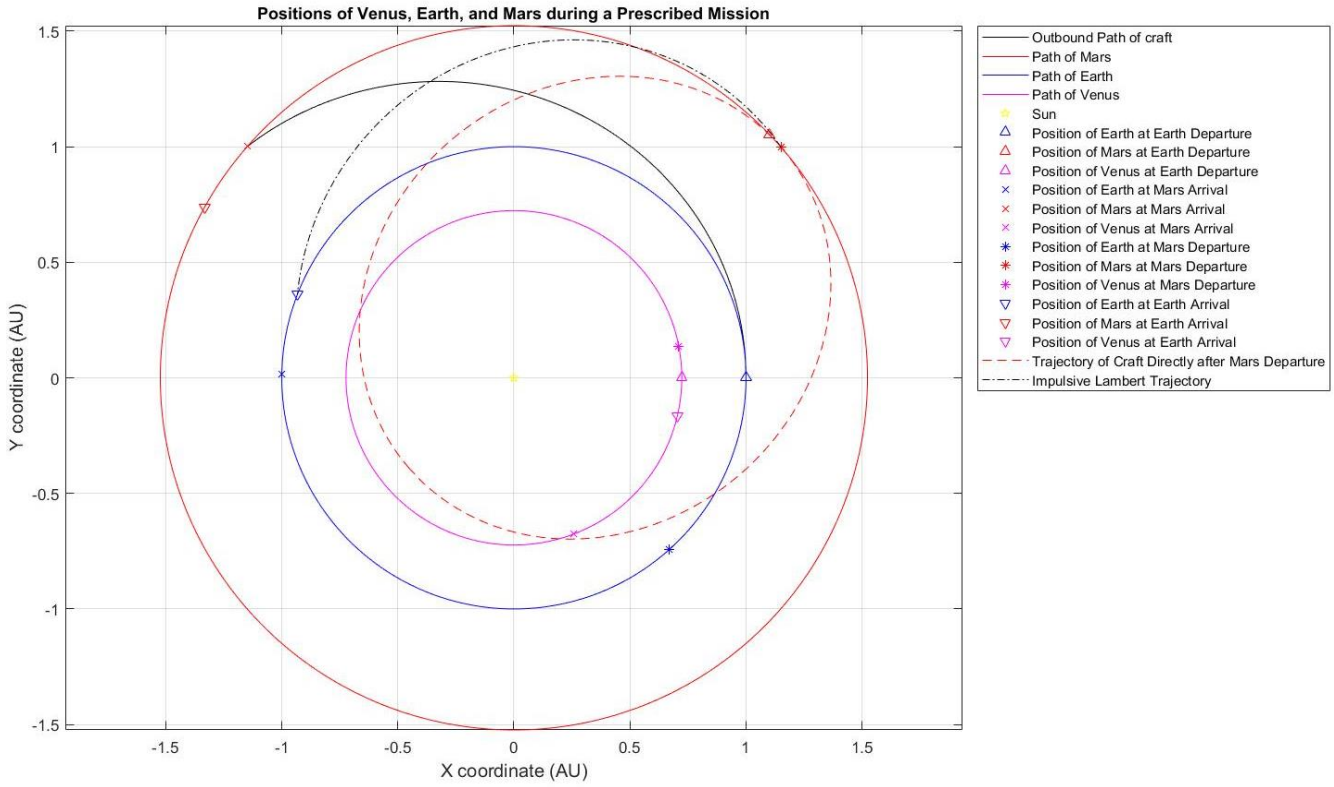

Figure 22: Long Duration Chemical Heliocentric Trajectory

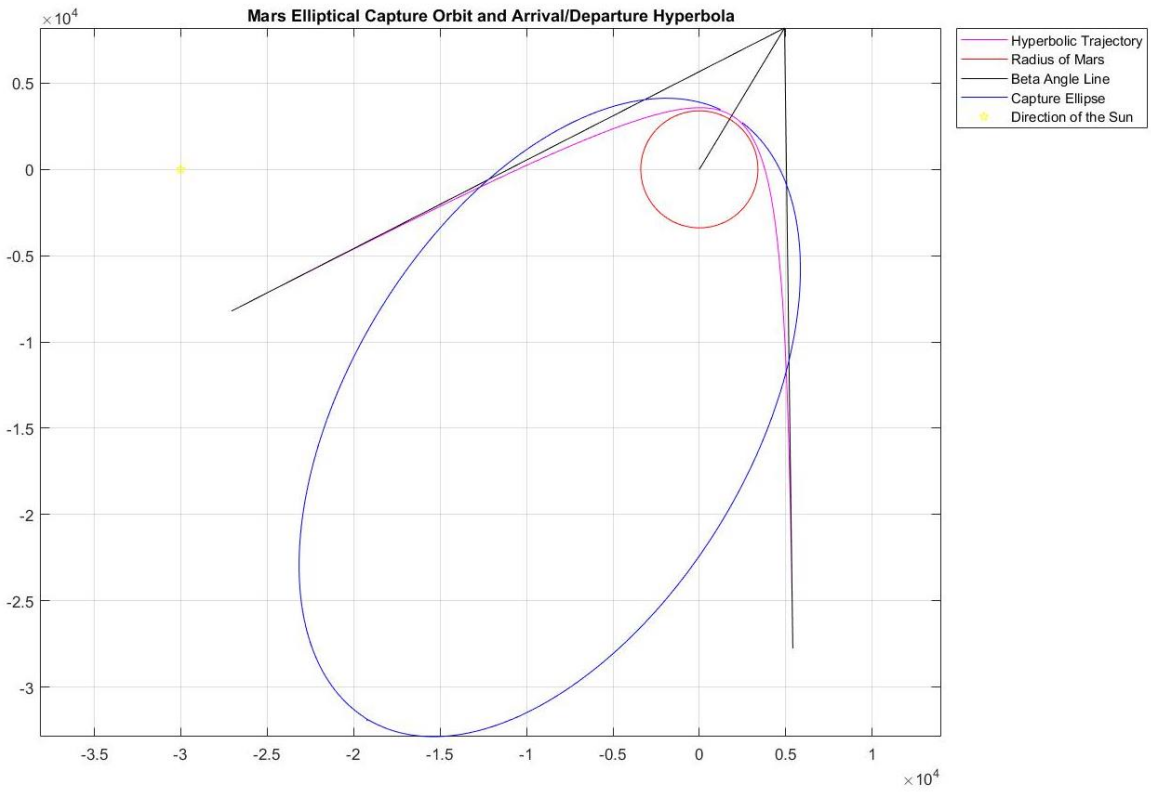

Figure 23: Long Duration Chemical Mars Arrival and Departure Trajectories

Table 1: Mission Parameters as Functions of Mission Propulsion Systems

Mission Propulsion System					
Mission Parameters	BNTEP	NTP - Mass	NTP - Time	Chemical - Mass	Chemical - Time
Mission Delta-V (km/s)	21.276 (12.149 - NTP) (9.127 - NEP)	14.975	24.115	12.863	24.115
Mission Duration (Days)	365.32	430.54	364.86	891.71	364.86
Earth Departure Mass (kg)	300,000	302,337	764,617	300,670	14,880,544

Chapter 4: Discussion

When comparing the results of the BNTEP mission with those of the equal time NTP and chemical propulsion missions, the lower delta-v value required by the BNTEP mission is intriguing. One possible reason for this difference is that when using an NEP system to reach Mars intercept, it's possible to specify the final transverse and radial velocities of the spacecraft, and in this case, those were 22.591 and 7.152 km/s, respectively. This ability allows the craft to reach any reasonable set of desired boundary conditions as it reaches Mars, and when planning to enter Martian orbit, slowing down beforehand is a possible way to reduce the delta-v values necessitated by MOI and Mars departure. By comparison, utilizing a single NTP or chemical burn at Earth to reach Mars in the same amount of time yields only one fixed final set of boundary conditions, which may make it more challenging to decelerate and accelerate at Mars.

A second possible cause for this difference in the delta-v values is due to the Oberth Effect. When using a continuous burn to reach a destination, NEP to and from Mars in this case, the propulsion system gains efficiency in accelerating, the closer it is to the bottom of its current gravity well. Here, that gravity well would be the sun, and for the given thrust and burn durations, since more delta-v is gained from the NEP system while returning to Earth than leaving it (the craft has a smaller mass), the Oberth Effect works in the spacecraft's favour, as it accelerates near the Sun after dropping within Earth's heliocentric orbit. (Refer to Figures 7-11). As for the NTP and chemical propulsion systems, both impulsively burn at the distance of Mars' orbit to return to Earth, thus not receiving the same benefit from the Oberth Effect as the BNTEP system.

Of course, a reduced delta-v is significant when concerned with the initial mass of the BNTEP spacecraft, but also having higher efficiency engines to impart that delta-v is significant as well. Utilizing the NEP system aboard the BNTEP spacecraft promises to reduce a considerable amount of mass from the craft at departure from LEO. In this specific case, the mission employing the BNTEP system gained 9.127 km/s of its 21.276 km/s total delta-v from its NEP system, and when using an engine with more than five times the efficiency (5000s for NEP vs. 950s for NTP), the necessary fuel mass is altered in an exponential way. This is the reason that the BNTEP system can complete the mission in 365.32 days with only an initial LEO mass of 300,000 kg. On the other hand, exclusively utilizing the NTP system required not only more delta-v but also wasn't as efficient when imparting that delta-v upon the spacecraft. Therefore, the mission required an initial LEO mass of 764,617 kg to complete the mission in 364.86 days. This trend continues as the propulsion system becomes more inefficient, and when replacing the NTP system with a chemical one (Isp ~ 450s), the initial LEO mass becomes 14,880,544 kg to impart the same 24.115 km/s worth of delta-v and complete the trip within the same 364.86 days.

Another result that deserves deeper analysis is the burn times required during key portions of the mission, as for both NTP and chemical propulsion, engine life is a factor. For NTP, engine life is partially due to the fact that over its lifetime, daughter products from the reactor self-pollute the core, leading to additional neutron absorption and decreasing energy production.

Overall, the craft with the BNTEP system fired its NTP engines for 101 minutes, which is within the estimated engine life of 2 hours or less for NTP engines (Borowski et al., 2012). For the faster standalone NTP mission, the engine burn time becomes 333 minutes, with some burns lasting over 2 hours. Such a burn time is outside of the estimated range for NTP engines and undermines the plausibility of such a mission. On the other hand, the slower, standalone NTP mission utilizes its NTP engines for 114 minutes and gives greater support for the mission's plausibility. Of course, the short duration chemical propulsion mission behaves in a similar way as the short duration NTP mission, and its overall burn time is 159.45 minutes. For the RS-25 engine, such a duration is far outside of the longest, continuous burn of 8.5 minutes utilized aboard the Space Shuttles, and therefore severely undermines the reality of this mission (Aerojet Rocketdyne). Finally, the long duration chemical mission proves itself to be plausible as its overall burn time is 5.11 minutes, well within the engine lifetime defined above.

These results are significant when concerned with initial spacecraft mass and total trip time and provide solid evidence to support a claim that higher efficiency engines should be used where possible to reduce fuel mass. However, high thrust engines are still necessary to quickly escape the spheres of influence of Earth and Mars, otherwise using only an NEP system would require considerable amounts of time to spiral out of these spheres of influence, and it's in this way that a BNTEP system is the most promising when shorter duration missions are desired.

Chapter 5: Conclusions and Future Work

Long duration spaceflight poses considerable health risks to humans, and while some technologies and practices have been developed to limit these threats, perhaps the simplest method would be to shorten overall mission duration. As has been demonstrated in this paper, utilizing a BNTEP system could allow crewed spacecraft to reach destinations faster and more efficiently than ever before, opening new mission possibilities that are safer and more economical in terms of both human cost and monetary cost.

However, assumptions were made in this paper than may have limited its results. Inclinations, eccentricities, and arguments of periapsis of Earth and Mars were not considered, and all were assumed to have circular, co-planer orbits; future studies should take these into account. Additionally, future papers could calculate astronaut radiation exposure as a function of distance and time relative to the sun, possibly yielding insightful data on how a BNTEP mission would affect crew radiation exposure.

References

Aerojet Rocketdyne, RS-25 Engine, https://www.rocket.com/rs-25-engine, (accessed 12/7/18).

D. Arney, K. Earle, B. Cirillo, C. Jones, J. Klovstad, M. Grande, The Impact of Mission Duration on a Mars Orbital Mission, NASA Archives (2017).

S.K. Borowski, "Bimodal" Nuclear Thermal Rocket (BNTR) Propulsion for Future Human Mars Exploration Missions, NASA Archives (2003).

S.K. Borowski, D.R. McCurdy, T.W. Packard, Nuclear Thermal Propulsion (NTP): A Proven Growth Technology for Human NEO / Mars Exploration Missions, NASA Archives (2012).

J.A. Castro Nieto, J. Del Valle, C. Martinez, A. Rivera, J. Oguilve, C.S. Olsen, M. Giambusso, M.D. Carter, J.P. Squire, F.R. Chang Diaz, VASIMR® VX-CR Experiment: Status, Diagnostics and Plasma Plume Characterization, 33rd International Electric Propulsion Conference (2013).

D. Eagle, Two-dimensional, Low-thrust Earth-to-Mars Trajectory Analysis with MATLAB. 8 April 2016, https://www.mathworks.com/matlabcentral/fileexchange/42374-two-dimensional-low-thrust-earth-to-mars-trajectory-analysis-with-matlab, (accessed 09/17/18).

D. Eagle, Lambert's Problem. 9 July 2014, https://www.mathworks.com/matlabcentral/fileexchange/39530-lambert-s-problem, (accessed 10/06/18).

H.P. Gerrish, Nuclear Thermal Propulsion Ground Test History: The Rover/NERVA Program, NASA Archives (2014).

P. E. Gill, W. Murray, M. A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, SIAM Review 47 (2005), 99-131.

M. Houts, S. Mitchell, K. Aschenbrenner, M. Van Dyke, H. Gerrish, Space Fission Propulsion and Power, NASA Archives (2014).

M. Kelly, OptimTraj -- Trajectory Optimization Library. 12 December 2015, https://www.mathworks.com/matlabcentral/fileexchange/54386-optimtraj-trajectory-optimization-library, (accessed 09/23/18).

D.R. McCurdy, S.K. Borowski, L.M. Burke, T.W. Packard, A Crewed Mission to Apophis Using a Hybrid Bimodal Nuclear Thermal Electric Propulsion (BNTEP) System, NASA Archives (2014).

Appendix A: BNTEP MATLAB Script

```
%Justin Clark (OSU .2740)

%%***PART 1 - EARTH DEPARTURE***%

%Justin Clark (.2740)
%Earth Departure Simulation


close all;
clear;
clc;

%define our initial mass of the craft
bntep_mass(1) = 310000; %kg

%define our desired payload mass... this will be the final mass we want to
%arrive at earth

ntp_mass = 4000; %kg
vasimr_mass = 620; %kg
habitat_mass = 30000; %kg - consumables, spares, habitat, humans for ~350-400
days
reentry_capsule = 10000; %kg
mass_payload = (3*ntp_mass + 6*vasimr_mass + habitat_mass +
reentry_capsule)*1.25; %kg, add in a 25% margin of error

%Define our constants for later use
re = 6378;
rm = 3389;
alt_e = 400; %LEO altitude of 400km above the surface
u_e = 398600;
u_s = 1.3271244 * 10^11;
u_m = 42648.65;
dist_earth = 149.6 * 10^6; %km
dist_mars = 227.9 * 10^6; %km
orbit_period_e = 365.256; %days
orbit_period_m = 687; %days

%find the distance from the sun at departure
dist_dep = dist_earth + alt_e;

%find periapsis of the earth
rp_e = alt_e + re;
%next, we need to the find the necessary delta-v to get from Earth orbit to
%the desired v infinity after leaving earth's SoI
v_inf = 4; %km/s
v_earth_park = (u_e/(re+alt_e))^.5;
v_earth_esc = (u_e*((2/rp_e) + ((v_inf^2)/u_e)))^.5;
bntep_delta_v(1) = v_earth_esc - v_earth_park;
fprintf('\nThe required Delta-V to get to the desired hyperbolic excess
velocity from LEO is %.4f km/s.\n',bntep_delta_v(1));

bntep_mass(2) = bntep_mass(1)*exp((-bntep_delta_v(1)*1000)/(9.81*950));

%now find the semi-major axis, eccentricity, and angular momentum of the
```

```
%transfer orbit and print the results

v_earth = ((u_s/dist_earth))^.5;
trans_h = (v_inf+v_earth)*dist_earth;
trans_a = (dist_earth + dist_mars)/2;
trans_e = ((-1*((trans_h^2)/(trans_a*u_s))) + 1)^.5;
%fprintf('\nFor the Hohmann transfer ellipse,the angular momentum is %.4f
km^2/s, the semi-major axis is %.4f km, and the eccentricity is %.4f
.\n',trans_h,trans_a,trans_e);

%now find the semi-major axis, eccentricity, and angular momentum of the
%hyperbolic departure and print the results

hyp_h = v_earth_esc*rp_e;
hyp_e = ((hyp_h^2)/(rp_e*u_e)) - 1;
hyp_a = ((hyp_h^2)/u_e)*(1/(1 - hyp_e^2));
theta_lim = 2*asind(1/hyp_e);
beta = (180 - theta_lim)/2;
%now plot both the hyperbolic departure orbit

%for the hyperbolic path, we can write a function of the path of the orbit
hyp_b = hyp_a*tand(beta);
hyp_fun = @(x) (-1*(hyp_b^2)*(1 - (((-1*x+(-1*hyp_a +
rp_e))^2)/(hyp_a^2))))^.5;

t = linspace(-15000,rp_e,1000);

%get the data points for the top and bottom of the ellipse, and delete any
%weird data points
for j=1:1:2000
    if j<1001
        x(j) = t(j);
        y(j) = hyp_fun(t(j));
    else
        x(j) = t(j-1000);
        y(j) = -1*hyp_fun(t(j-1000));
    end
    if j == 1001
        y(j) = NaN;
    end

end


%plot the hyperbolic escape trajectory, the radius of the Earth, and the
%beta angle line

poly = polyfit([x(1),x(2)],[y(1),y(2)],1);
zero_location_x = -1*poly(2)*(1/poly(1));

%find the beta angle needed for the escape trajectory
beta = atand(y(1)/(zero_location_x - x(1)));
fprintf('\nThe beta angle required for the departure burn is %.4f
degrees.\n',beta);

beta = 90 - 42.9203;
figure(1);
plot(x*cosd(-beta) - y*sind(-beta),y*cosd(-beta) + x*sind(-beta),'r-');
```

```
hold on;
axis equal;
grid on;

rad = linspace(0,2*pi,1000);
plot(re*cos(rad),re*sin(rad),'b-');
%,[x(1),zero_location_x,0,zero_location_x,x(1)]*cosd(-beta) - [y(1),0,0,0,-
y(1)]*sind(-beta),[y(1),0,0,0,-y(1)]*cosd(-beta) +
[x(1),zero_location_x,0,zero_location_x,x(1)]*sind(-beta),'k-');
plot(-30000,0,'kp');
title('Earth Escape Trajectory with a V\infty = 4.0 km/s');
xlabel('X coordinate (km)');
ylabel('Y coordinate (km)');
legend('Hyperbolic Trajectory','Radius of the Earth','Direction of the
Sun','location','bestoutside');



%%***PART 2 - CONSTANT THRUST OUTBOUND ***%

% dto_trap_64bit.m    July 6, 2014

% trajectory optimization using direct transciption and collocation

% trapezoid collocation, Chebyshev-Gauss-Lobatto
% node placement, and 64 bit SNOPT algorithm (Mar 17, 2014 version)

% Bryson-Ho example - maximize final radial distance

% Orbital Mechanics with MATLAB

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%simulation time
time_norm_test = 2.3;
v_final = .7581; %sqrt(1.0 / 1.524);
u_final = .24;

%test case and begin looping
mars_test_value = 0;

while mars_test_value ~= 1

    clearvars -except mass_payload time_norm_test mars_test_value v_final
u_final v_inf bntep_mass bntep_delta_v number_vas

global nc_total nc_defect ndiffeq nnodes nlp_state ncv

global nlp_control tarray acc beta

%state initial variables (spacecraft design variables)

% initial mass (kilograms)
mass0 = bntep_mass(2);

% propellant flow rate (kilograms/day)
number_vas = 6;
mdot = number_vas*8.807;
```

```
% thrust (newtons)
thrmag = number_vas*6;

%v_inf leaving earth
v_inf_earth = v_inf;
vel_trans_norm = (v_inf_earth+30)/30;


pi2 = 2.0 * pi;

% conversion factor - radians to degrees

rtd = 180.0 / pi;

% astronomical unit (kilometers)

aunit = 149597870.691;

% gravitational constant of the sun (km**3/sec**2)

xmu = 132712441933.0;

% time conversion factor

tcf = sqrt(aunit^3 / xmu) / 86400.0;

% normalized propellant flow rate

beta = (mdot / mass0) * tcf;

% number of differential equations

ndiffeq = 3;

% number of control variables

ncv = 1;

% number of discretization nodes

nnodes = 50;

% number of state nlp variables

nlp_state = ndiffeq * nnodes;

% number of control nlp variables

nlp_control = ncv * nnodes;

% total number of nlp variables

nlpv = nlp_state + nlp_control;

% number of state vector defect equality constraints

nc_defect = nlp_state - ndiffeq;

% number of auxilary equality constraints (boundary conditions)
```

```
nc_aux = 2;

% total number of equality constraints

nc_total = nc_defect + nc_aux;


% normalized thrust acceleration

acc = 0.001 * (thrmag / mass0) / (xmu / aunit^2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% initial and final times and states
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% initial simulation time

tinitial = 0.0;

% final simulation time

tfinal = time_norm_test; %58.129 days/1 unit of time (about 365/(2pi))

% define state vector at initial time

xinitial(1) = 1.0;

xinitial(2) = 0.0;

xinitial(3) = vel_trans_norm;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create time values at nodes
% (Chebyshev-Gauss-Lobatto)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[tarray, w] = cgl(nnodes, tinitial, tfinal);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% upper and lower bounds for nlp state variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 1:1:nlp_state

    xlb(i) = -2.0;

    xub(i) = +2.0;

end

% constrain initial boundary conditions
% to be the initial state vector

for i = 1:1:ndiffeq

    xlb(i) = xinitial(i);

    xub(i) = xinitial(i);
```

```
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% upper and lower bounds for nlp control variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = nlp_state + 1:1:nlpv

    xlb(i) = -pi;

    xub(i) = +pi;

end

xlb = xlb';

xub = xub';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% initial guess for nlp state variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

j = 0;

for i = 1:1:nlp_state

    j = j + 1;

    xg(i) = xinitial(j);

    if (j == ndiffeq)
        j = 0;
    end

end

lgflag = 1;

if (lgflag == 1)

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % compute linear initial guess for state variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % initial conditions

    xi1 = xinitial(1);

    xi2 = xinitial(2);

    xi3 = xinitial(3);

    % final conditions

    xf1 = 1.525;

    xf2 = u_final;
```

```
    xf3 = v_final;
%
%     xlb(148) = xf1;
%     xub(148) = xf1;
    xlb(149) = xf2;
    xub(149) = xf2;
    xlb(150) = xf3;
    xub(150) = xf3;
    % initial and final times

    ti = tinitial;

    tf = tfinal;

    % compute linear guesses

    xg1 = dto_guess(ti, tf, xi1, xf1, nnodes);

    xg2 = dto_guess(ti, tf, xi2, xf2, nnodes);

    xg3 = dto_guess(ti, tf, xi3, xf3, nnodes);

    % load initial guess array

    j = 1;

    for i = 1: nnodes

        xg(j) = xg1(i);

        xg(j + 1) = xg2(i);

        xg(j + 2) = xg3(i);

        j = j + 3;

    end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% initial guess for nlp control variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = nlp_state + 1:1:nlpv

    xg(i) = 0.0;

end

% transpose initial guess

xg = xg';

% define lower and upper bounds on objective function

flow(1) = 1.51;
```

```
fupp(1) = 1.53;

% define lower and upper bounds on defects

for i = 1:1:nc_defect

    flow(i + 1) = 0.0;

    fupp(i + 1) = 0.0;

end

% define lower and upper bounds on final boundary conditions

%for i = 1:1:nc_aux

    flow(1 + nc_defect + 1) = xf2;

    fupp(1 + nc_defect + 1) = xf2;

    flow(2 + nc_defect + 1) = xf3;

    fupp(2 + nc_defect + 1) = xf3;

%end

flow = flow';

fupp = fupp';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% solve direct transcription problem
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% set SNOPT file options

%snprintfile('dto_trap.out');

snsummary('dto_trap.sum');

snscreen('on');

xmul = zeros(200, 1);

xstate = zeros(200, 1);

fmul = zeros(150, 1);

fstate = zeros(150, 1);

%[x, f, inform, xmul, fmul] = snopt(xg, xlb, xub, flow, fupp,
'dtofunc_trap');

[x, f, inform, xmul, fmul] = snopt(xg, xlb, xub, xmul, xstate, ...
    flow, fupp, fmul, fstate, 'dtofunc_trap');

%snprintfile('off');

snsummary('off');
```

```
% print results

fprintf('\n\n         program dto_trap \n');

fprintf('\n    trajectory optimization using');

fprintf('\n direct transcription and collocation \n\n');

fprintf ('\ninitial state vector \n');

fprintf ('\n radius              = %12.8f \n', x(1));
fprintf ('\n radial velocity     = %12.8f \n', x(2));
fprintf ('\n transverse velocity = %12.8f \n', x(3));

fprintf ('\n\nfinal state vector \n');

fprintf ('\n radius               = %12.8f \n', x(nlp_state - 2));
fprintf ('\n radial velocity      = %12.8f \n', x(nlp_state - 1));
fprintf ('\n transverse velocity  = %12.8f \n\n', x(nlp_state));

if x(nlp_state - 2) > 1.525
    mars_test_value = 1;
end

%%%%%%%%%%%%%%%%%%%%%%%%%
% create trajectory plots
%%%%%%%%%%%%%%%%%%%%%%%%%

npts = 0;

j = 1;

for i = 1:1:nnodes

    npts = npts + 1;

    % simulation time

    xplot(i) = tarray(i) * tcf;

    % radius

    yplot2(i) = x(j);

    % radial velocity

    yplot3(i) = x(j + 1);

    % transverse velocity

    yplot4(i) = x(j + 2);

    j = j + 3;

end

% polar angle (radians)

theta = cumtrapz(tarray, yplot4./yplot2);
```

```
j = 0;

for i = nlp_state + 1:1:nlpv

    j = j + 1;

    % alpha (degrees)

    yplot1(j) = rtd * x(i);

end

% plot results

figure(2);
subplot(4,1,1);
plot(xplot, yplot1, '-*');

title('Trajectory Parameters of the Outbound Path');

ylabel('\gamma (\circ)','FontSize', 12);

grid;

print -depsc -tiff -r300 control.eps

subplot(4,1,2);
plot(xplot, yplot2, '-*');

ylabel('r (AU)', 'FontSize', 12);

grid;

print -depsc -tiff -r300 rdistance.eps

subplot(4,1,3);

plot(xplot, 29.79*yplot3, '-*');

ylabel('U (km/s)', 'FontSize', 12);

grid;

print -depsc -tiff -r300 vradial.eps

subplot(4,1,4);

plot(xplot, 29.79*yplot4, '-*');

xlabel('Simulation Time (days)', 'FontSize', 12);

ylabel('V (km/s)', 'FontSize', 12);

grid;

print -depsc -tiff -r300 vtransverse.eps

%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create trajectory display
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(3); clf;

hold on;

axis([-1.7 1.7 -1.6 1.6]);

% create array of polar angles (radians)

t = 0: pi / 50.0: 2.0 * pi;

% plot and label transfer orbit

plot(yplot2.* cos(theta), yplot2.*sin(theta), 'k-');

% label plot and axes

xlabel('X coordinate (AU)', 'FontSize', 12);

ylabel('Y coordinate (AU)', 'FontSize', 12);

grid;

axis equal;

zoom on;

print -depsc -tiff -r300 trajectories.eps

time_norm_test = time_norm_test + .05;
end


fprintf('\n\n***SOLUTION FOUND***\n');

bntep_mass(3) = bntep_mass(2) - time_norm_test*tcf*mdot;
bntep_delta_v(2) = (9.81/1000)*5000*log(bntep_mass(2)/bntep_mass(3));;
time_outbound = time_norm_test*(365/(2*pi));


%***PART 3 - HELIOCENTRIC PLOT***%

%Plot positions of Venus, Earth, and Mars as time progresses, with initial
%angles given

aunit = 149597870.691;

%Define our constants for later use
dist_venus = 108.2 *10^6 / aunit;
dist_earth = 149.6 * 10^6 / aunit;
dist_mars = 227.9 *  10^6 / aunit;
orbit_period_e = 365.256;
orbit_period_m = 687;
orbit_period_v = 225;

%now calculate our rotation rates and the synodic period and print the
%results
n1 = (2*pi*(1/orbit_period_e)); %rad/day
```

```
n2 = (2*pi*(1/orbit_period_m));
n3 = (2*pi*(1/orbit_period_v));
T_sys = 2*pi*(1/abs(n2 - n1));

%fprintf('\nWhat is the angle between Earth and Venus in degrees? (Positive
indicates Venus ahead)')
venus_angle = 0; %(pi/180)*input('\n');
%fprintf('\nWhat is the angle between Earth and Mars in degrees? (Positive
indicates Mars ahead)')
mars_angle = theta(50) - n2*time_outbound; %(pi/180)*input('\n');
earth_angle = 0;

%fprintf('\nHow long will the outbound leg to Mars take? (In Days)');
outbound_time = time_outbound; %input('\n');
%fprintf('\nHow long will the craft stay at Mars? (In Days)');
stay_time = 20; %input('\n');
fprintf('\nHow long will the inbound leg to Earth take? (In Days)');
inbound_time = 203; %input('\n');

fprintf('\nThe synodic period between Earth and Mars is %.4f days or %.4f
years.\n',T_sys,T_sys/orbit_period_e);

%now plot the ellipse with the orbits of Jupiter and Earth, and place
%points indicating the locations of Earth and Jupiter upon arrival

figure(3);
hold on;
axis equal;
grid on;

%plot the circles
rad = linspace(0,2*pi,1000);
plot(dist_mars*cos(rad),dist_mars*sin(rad),'r-');
plot(dist_earth*cos(rad),dist_earth*sin(rad),'b-');
plot(dist_venus*cos(rad),dist_venus*sin(rad),'m-');

%plot the points of the planets and the sun
dep_1_e_x = dist_earth;
dep_1_e_y = 0;

dep_1_m_x = dist_mars*cos(mars_angle);
dep_1_m_y = dist_mars*sin(mars_angle);

dep_1_v_x = dist_venus*cos(venus_angle);
dep_1_v_y = dist_venus*sin(venus_angle);

arr_1_e_x = dist_earth*cos(earth_angle + n1*outbound_time);
arr_1_e_y = dist_earth*sin(earth_angle + n1*outbound_time);

arr_1_m_x = dist_mars*cos(mars_angle + n2*outbound_time);
arr_1_m_y = dist_mars*sin(mars_angle + n2*outbound_time);

arr_1_v_x = dist_venus*cos(venus_angle + n3*outbound_time);
arr_1_v_y = dist_venus*sin(venus_angle + n3*outbound_time);

dep_2_e_x = dist_earth*cos(earth_angle + n1*(outbound_time + stay_time));
dep_2_e_y = dist_earth*sin(earth_angle + n1*(outbound_time + stay_time));
```

```
dep_2_m_x = dist_mars*cos(mars_angle + n2*(outbound_time + stay_time));
dep_2_m_y = dist_mars*sin(mars_angle + n2*(outbound_time + stay_time));

dep_2_v_x = dist_venus*cos(venus_angle + n3*(outbound_time + stay_time));
dep_2_v_y = dist_venus*sin(venus_angle + n3*(outbound_time + stay_time));

arr_2_e_x = dist_earth*cos(earth_angle + n1*(outbound_time + stay_time +
inbound_time));
arr_2_e_y = dist_earth*sin(earth_angle + n1*(outbound_time + stay_time +
inbound_time));

arr_2_m_x = dist_mars*cos(mars_angle + n2*(outbound_time + stay_time +
inbound_time));
arr_2_m_y = dist_mars*sin(mars_angle + n2*(outbound_time + stay_time +
inbound_time));

arr_2_v_x = dist_venus*cos(venus_angle + n3*(outbound_time + stay_time +
inbound_time));
arr_2_v_y = dist_venus*sin(venus_angle + n3*(outbound_time + stay_time +
inbound_time));

plot(0,0,'kp',dep_1_e_x,dep_1_e_y,'b^',dep_1_m_x,dep_1_m_y,'r^',arr_1_e_x,arr
_1_e_y,'bx',arr_1_m_x,arr_1_m_y,'rx',dep_2_e_x,dep_2_e_y,'b*',dep_2_m_x,dep_2
_m_y,'r*',arr_2_e_x,arr_2_e_y,'bv',arr_2_m_x,arr_2_m_y,'rv');
title('Interplanetary BNTEP Mission Trajectory');


%***PART 4 - MARS ARRIVAL AND DEPARTURE***%

%Justin Clark (OSU .2740)
%Mars Arrival and Departure with Burns

clearvars -except mass_payload time_norm_test v_final u_final v_inf
mars_angle n1 n2 n3 outbound_time stay_time inbound_time aunit bntep_mass
bntep_delta_v number_vas

u_spacecraft = u_final; %relative to earths speed of 29.658 km/s
v_spacecraft = v_final;

%Mars speed is 24.077 km/s

mu_mars = 42828.37; % km^3/s^2
v_mars = 24.077/29.658; %relative to earth
r_mars = 3389; %km

v_spi = (((v_spacecraft - v_mars)^2)+(u_spacecraft)^2)^.5; %relative to earth

if v_spacecraft > v_mars
theta_i = 180 - atand(u_spacecraft/(v_spacecraft - v_mars)); %deg
else
theta_i = abs(atand(u_spacecraft/(v_spacecraft - v_mars))); %deg
end

%capture orbit parameters

rp = 250 + r_mars; %km
ra = 33813 + r_mars; %km
a_cap = (rp + ra)/2; %km
```

```
e_cap = 1 - (rp/a_cap);
vp_cap = ((mu_mars*(1+e_cap))/rp)^.5; %km/s

%hyperbola parameters

turning_angle = theta_i; %deg
e_hyp = 1/(sind(turning_angle/2));
impact_parameter = rp*((1 + ((2*mu_mars)/(rp*((v_spi*29.658)^2))))^.5); %km
a_hyp = -impact_parameter/(((e_hyp^2)-1)^.5); %km
vp_hyp = (((v_spi*29.658)^2) + ((2*mu_mars)/(rp)))^.5; %km/s

%delta-v needed at periapsis to get into capture orbit
delta_v_arrival = vp_hyp - vp_cap; %km/s
bntep_delta_v(3) = 2*delta_v_arrival;

bntep_mass(4) = bntep_mass(3)*exp((-2*delta_v_arrival*1000)/(9.81*950));

fprintf('\nThe aim distance at Mars should be %.3f km.\n',impact_parameter);
fprintf('\nThe required delta-v to enter the elliptical capture orbit from
the hyperbola is -%.3f km/s.',delta_v_arrival);
fprintf('\nThe total delta-v for arrival at and departure from Mars is
therefore %.3f km/s.\n',2*delta_v_arrival);

%if leaving mars using the same hyperbola, calculate the v_inf leaving
v_spf = v_spi; %rel. to mars, but in the negative transverse direction

v_inf_mars_departure = v_spf*29.658;

%%***PLOT***%%
%now that all of the necessary parameters have been found, plot the
%hyperbolic and elliptical orbits

%find the beta angle needed for the escape trajectory
theta_lim = acosd(-1/e_hyp);
beta = 180 - theta_lim;
fprintf('\nThe beta angle required for the departure burn is %.4f
degrees.\n',beta);
fprintf('\nThe hyperbolic excess velocity leaving Mars is -%.3f km/s (u_v)
and 0.000 km/s (u_s).\n',v_spf*29.658);

%now plot both the hyperbolic departure orbit

%for the hyperbolic path, we can write a function of the path of the orbit
hyp_b = a_hyp*tand(beta);
hyp_fun = @(x) (-1*(hyp_b^2)*(1 - (((-1*x+(-1*a_hyp +
rp))^2)/(a_hyp^2))))^.5;

t = linspace(-21000,rp,1000);

%get the data points for the top and bottom of the ellipse, and delete any
%weird data points
for j=1:1:2000
    if j<1001
        x(j) = t(j); %#ok<*SAGROW>
        y(j) = hyp_fun(t(j));
    else
        x(j) = t(j-1000);
        y(j) = -1*hyp_fun(t(j-1000));
```

```
    end
    if j == 1001
        y(j) = NaN;
    end

end

%plot the hyperbolic escape trajectory, the radius of Mars, and the
%beta angle line

poly = polyfit([x(1),x(2)],[y(1),y(2)],1);
zero_location_x = -1*poly(2)*(1/poly(1));

figure(4);
plot(cosd(-beta - 270)*x - sind(-beta - 270)*y,sind(-beta - 270)*x + cosd(-
beta - 270)*y,'m-');
hold on;
axis equal;
grid on;
rad = linspace(0,2*pi,1000);
plot(r_mars*cos(rad),r_mars*sin(rad),'r-'); %,cosd(-beta -
270)*[x(1),zero_location_x,0,zero_location_x,x(1)] - sind(-beta -
270)*[y(1),0,0,0,-y(1)],sind(-beta -
270)*[x(1),zero_location_x,0,zero_location_x,x(1)] + cosd(-beta -
270)*[y(1),0,0,0,-y(1)],'k-');
title('Mars Elliptical Capture Orbit and Arrival/Departure Hyperbola');
xlabel('X coordinate (km)');
ylabel('Y coordinate (km)');

%define a function to plot the ellipse
b_cap = ((a_cap^2)*(1 - (e_cap^2)))^.5;
ellipse_fun = @(x) ((b_cap^2)*(1 - (((x+(a_cap - rp))^2)/(a_cap^2))))^.5;
t = linspace(-37300,rp,1000);

%get the data points for the top and bottom of the ellipse, and delete any
%weird data points
for j=1:1:2000
    if j<1001
        x(j) = t(j);
        y(j) = ellipse_fun(t(j));
    else
        x(j) = t(j-1000);
        y(j) = -1*ellipse_fun(t(j-1000));
    end
    if y(j) == 0
        y(j) = NaN;
    end

end

%now plot the ellipse
figure(4);
hold on;
plot(cosd(-beta - 270)*x - sind(-beta - 270)*y,sind(-beta - 270)*x + cosd(-
beta - 270)*y,'b-');
plot(-30000,0,'kp');
```

```
legend('Hyperbolic Trajectory','Radius of Mars','Capture Ellipse','Direction
of the Sun','location','bestoutside');

%now plot the heliocentric ellipse after departing mars (before NEP
%burn)

x = [];
y = [];

global mars_departure_va mars_departure_h mars_departure_a mars_departure_e
mars_departure_rp mars_departure_theta dist_mars mars_velocity
u_s = 1.3271244 * 10^11;
dist_mars = 227.9 * 10^6; %km
mars_velocity = 24.07; %km/s
mars_departure_va = mars_velocity - v_inf_mars_departure;
mars_departure_h = mars_departure_va * dist_mars;
mars_departure_a = 1/((2/dist_mars)-((mars_departure_va^2)/u_s));
mars_departure_e = ((-1*((mars_departure_h^2)/(mars_departure_a*u_s))) +
1)^.5;
mars_departure_rp = mars_departure_a*(1 - mars_departure_e);
mars_departure_theta = (mars_angle + n2*(outbound_time +
stay_time))*(180/pi); %deg

%define a function to plot the ellipse
mars_departure_b = (((mars_departure_a^2)*(1 -
(mars_departure_e^2)))^.5)/aunit;
ellipse_fun = @(x) ((mars_departure_b^2)*(1 - (((x+((mars_departure_a -
mars_departure_rp)/aunit))^2)/((mars_departure_a/aunit)^2))))^.5;
t = linspace(-dist_mars/aunit,mars_departure_rp/aunit,1000);

%get the data points for the top and bottom of the ellipse, and delete any
%weird data points
for j=1:1:2000
    if j<1001
        x(j) = t(j);
        y(j) = ellipse_fun(t(j));
    else
        x(j) = t(j-1000);
        y(j) = -1*ellipse_fun(t(j-1000));
    end
    if j == 1001
        y(j) = NaN;
    end

end

%now plot the ellipse
figure(3);
%plot(cosd(mars_departure_theta + 180)*x - sind(mars_departure_theta +
180)*y,sind(mars_departure_theta + 180)*x + cosd(mars_departure_theta +
180)*y,'r--');


%***PART 5 - IMPULSIVE LAMBERTS PROBLEM***%

% lambert2.m     December 16, 2012

% solution of the interplanetary Lambert problem
```

```
% Orbital Mechanics with Matlab

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global mu tbcoef

% define Astronomical Unit (kilometers)

% initialize two body propagator "twobody1"

tbcoef = 1;

% load planet name array

pdata = ['Mercury'; 'Venus  '; 'Earth  '; 'Mars   '; ...
      'Jupiter'; 'Saturn '; 'Uranus '; 'Neptune'; 'Pluto  '];

pname = cellstr(pdata);

% begin simulation

fprintf('\n            program lambert2\n');

fprintf('\n   < interplanetary lambert problem >\n\n');

fprintf('\ndeparture conditions\n');

delta_v_mag = 100000;
fprintf('\n  working ...\n');

taud = inbound_time;

tof = taud * 86400;

ip1 = 4;
ip2 = 3;

% compute orbital elements of departure and arrival planets

[ri, vi] = planet_altered_2(ip1,(n2*(outbound_time + stay_time))+mars_angle);

oevi = eci2orb1(mu, ri, vi);

[rf, vf] = planet_altered_2(ip2,n1*(outbound_time + stay_time +
inbound_time));

oevf = eci2orb1(mu, rf, vf);

% solve Lambert's problem

for i = 1:1:3

    sv1(i) = ri(i);

    sv1(i + 3) = vi(i);

    sv2(i) = rf(i);

    sv2(i + 3) = vf(i);
```

```
end

[vito, vfto] = glambert(mu, sv1, sv2, tof, 0);

oevtoi = eci2orb1(mu, ri, vito');

oevtof = eci2orb1(mu, rf, vfto');

% initial delta-v vector (kilometers/second)

dvi(1) = vito(1) - vi(1);
dvi(2) = vito(2) - vi(2);
dvi(3) = vito(3) - vi(3);

% final delta-v vector (kilometers/second)

dvf(1) = vf(1) - vfto(1);
dvf(2) = vf(2) - vfto(2);
dvf(3) = vf(3) - vfto(3);


% convert time and dates to strings

%[cdstr1, utstr1] = jd2str(jdate1);

%[cdstr2, utstr2] = jd2str(jdate2);

% print results

fprintf('\n\n         program lambert2\n');

fprintf('\n< interplanetary lambert problem >\n');

fprintf('\ndeparture planet        ');

disp(pname(ip1));

fprintf('\n\ntransfer time                %12.6f  days \n ', taud);

fprintf('\nheliocentric ecliptic orbital elements of the departure
planet\n');

oeprint2(mu, oevi);

fprintf('\n\nheliocentric ecliptic orbital elements of the transfer orbit
after the initial delta-v\n');

oeprint2(mu, oevtoi);

fprintf('\n\nheliocentric ecliptic orbital elements of the transfer orbit
prior to the final delta-v\n');

oeprint2(mu, oevtof);

fprintf('\n\nheliocentric ecliptic orbital elements of the arrival planet
\n');

oeprint2(mu, oevf);
```

```
fprintf('\n\ninitial delta-v vector and magnitude\n');

fprintf('\nx-component of delta-v      %12.6f  meters/second', 1000.0 *
dvi(1));

fprintf('\ny-component of delta-v      %12.6f  meters/second', 1000.0 *
dvi(2));

fprintf('\nz-component of delta-v      %12.6f  meters/second', 1000.0 *
dvi(3));

fprintf('\n\ndelta-v magnitude           %12.6f  meters/second\n', 1000.0 *
norm(dvi));

fprintf('\nenergy                      %12.6f  km^2/sec^2\n', norm(dvi) *
norm(dvi));

fprintf('\n\nfinal delta-v vector and magnitude\n');

fprintf('\nx-component of delta-v      %12.6f  meters/second', 1000.0 *
dvf(1));

fprintf('\ny-component of delta-v      %12.6f  meters/second', 1000.0 *
dvf(2));

fprintf('\nz-component of delta-v      %12.6f  meters/second', 1000.0 *
dvf(3));

fprintf('\n\ndelta-v magnitude           %12.6f  meters/second\n', 1000.0 *
norm(dvf));

fprintf('\nenergy                      %12.6f  km^2/sec^2\n', norm(dvf) *
norm(dvf));

%%%%%%%%%%%
% graphics %
%%%%%%%%%%%

deltat = 1;

% planet periods

[r1, v1] = planet_altered_2(ip1,(n2*(outbound_time + stay_time))+mars_angle);
% jdate1);

oev1 = eci2orb1(mu, r1, v1);

period1 = 2 * pi * oev1(1) * sqrt(oev1(1) / mu) / 86400;

[r2, v2] = planet_altered_2(ip2,n2*(outbound_time + stay_time +
inbound_time)); % jdate1);

oev2 = eci2orb1(mu, r2, v2);

period2 = 2 * pi * oev2(1) * sqrt(oev2(1) / mu) / 86400;

xve = oev1(1) / aunit;
```

```
if (oev2(1) > oev1(1))
   xve = oev2(1) / aunit;
end

% determine number of data points to plot

npts1 = fix(period1 / deltat);

npts2 = fix(period2 / deltat);

npts3 = taud/deltat; %fix((jdate2 - jdate1) / deltat);

[rti, vti] = orb2eci(mu, oevtoi);

% create departure planet orbit data points

for i = 0:1:npts1

   %jdate = jdate1 + i * deltat;

   [r1, v1] = planet_altered_2(ip1,(n2*(outbound_time + stay_time +
i*deltat))+mars_angle);

   x1(i + 1) = r1(1) / aunit;

   y1(i + 1) = r1(2) / aunit;

end

% compute last data point

[r1, v1] = planet_altered_2(ip1, (n2*(outbound_time + stay_time +
period1))+mars_angle);

x1(npts1 + 2) = r1(1) / aunit;

y1(npts1 + 2) = r1(2) / aunit;

% create arrival planet orbit data points

for i = 0:1:npts2

   %jdate = jdate1 + i * deltat;

   [r2, v2] = planet_altered_2(ip2, n1*(outbound_time + stay_time +
i*deltat));

   x2(i + 1) = r2(1) / aunit;

   y2(i + 1) = r2(2) / aunit;

end

% compute last data point

[r2, v2] = planet_altered_2(ip2,n1*(outbound_time + stay_time + period2));

x2(npts2 + 2) = r2(1) / aunit;
```

```
y2(npts2 + 2) = r2(2) / aunit;

% create transfer orbit data points

for i = 0:1:npts3

   tau = 86400 * i * deltat;

   [rft, vft] = twobody1(mu, tau, ri, vito');

   x3(i + 1) = rft(1) / aunit;

   y3(i + 1) = rft(2) / aunit;

end

% compute last data point

tau = 86400 * taud; %(jdate2 - jdate1);

[rft, vft] = twobody1(mu, tau, rti, vti);

x3(npts3 + 2) = rft(1) / aunit;

y3(npts3 + 2) = rft(2) / aunit;

% plot planet orbits and transfer trajectory

figure(3);
hold on;

%plot(x3,y3,'g-.');

% label plot and axes

xlabel('X coordinate (AU)', 'FontSize', 12);

ylabel('Y coordinate (AU)', 'FontSize', 12);

zoom on;

print -depsc -tiff -r300 lambert2.eps;


%***PART 6 - CONSTANT THRUST RETURN OPTIMIZATION***%

% MAIN - Minimum Time Boundary Value Problem
%
% Solve a minimum-time boundary value problem with simple dynamics (chain
% integrator) and limits on the state and control. Scalar trajectory.
%
% Here we will solve a scalar trajectory, where the position, velocities,
% and angle of position are states. The thrust angle will be the only
% control.

addpath ../../

%state initial variables (spacecraft design variables)
```

```
global acc_new beta_new

% initial mass (kilograms)
mass0 = bntep_mass(4);

% propellant flow rate (kilograms/day)
mdot = number_vas*8.807;

% thrust (newtons)
thrmag = number_vas*6;

%v_inf leaving mars
vel_trans_norm = ((-1*v_inf_mars_departure)+24.07)/30;

pi2 = 2.0 * pi;

% conversion factor - radians to degrees
rtd = 180.0 / pi;

% conversion factor - degrees to radians
dtr = pi / 180.0;

% astronomical unit (kilometers)
aunit = 149597870.691;

% gravitational constant of the sun (km**3/sec**2)
xmu = 132712441933.0;

% time conversion factor
tcf = sqrt(aunit^3 / xmu) / 86400.0;

% normalized propellant flow rate
beta_new = (mdot / mass0) * tcf;

% normalized thrust acceleration
acc_new = 0.001 * (thrmag / mass0) / (xmu / aunit^2);


% Kinematic Limits:
rLim = [0, 1.6]; % position
vLim = [.5, 2]; % transverse velocity
uLim = [-.6, .6]; % radial velocity
thetaLim = [0, 2*pi]; % angle position
gammaLim = [-pi, pi]; % thrust angle

% Boundary value problem:
rBegin = 1.524;  % initial state
vBegin = vel_trans_norm;
uBegin = 0;
thetaBegin = n2*(outbound_time + stay_time) + mars_angle;
rFinal = 1;  % final state
vFinal = 1;
uFinal = .4;
thetaFinal = (n1*(outbound_time + stay_time + inbound_time));

% User-defined dynamics and objective functions
problem.func.dynamics = @(t,x,u)( scalarChainIntegrator_NEP_ATTEMPT(t,x,u) );
```

```
%problem.func.bndObj = @(t0,x0,tF,xF)( tF - t0 ); % minimum time  -- primary
objective
%problem.func.pathObj = @(t,x,u)( 0.001*u.^2 ); %minimum jerk  --
regularization

% Problem boundsTime
problem.bounds.initialTime.low = 0;
problem.bounds.initialTime.upp = 0;
problem.bounds.finalTime.low = inbound_time/tcf;
problem.bounds.finalTime.upp = inbound_time/tcf;

problem.bounds.state.low = [rLim(1); vLim(1); uLim(1); thetaLim(1)];
problem.bounds.state.upp = [rLim(2); vLim(2); uLim(2); thetaLim(2)];
problem.bounds.initialState.low = [rBegin; vBegin; uBegin; thetaBegin];
problem.bounds.initialState.upp = [rBegin; vBegin; uBegin; thetaBegin];
problem.bounds.finalState.low = [rFinal; vFinal - .2; uFinal - .2;
thetaFinal];
problem.bounds.finalState.upp = [rFinal; vFinal + .2; uFinal + .2;
thetaFinal];

problem.bounds.control.low = gammaLim(1);
problem.bounds.control.upp = gammaLim(2);

% Guess at the initial trajectory
problem.guess.time = [0,4];
problem.guess.state = [[rBegin; vBegin; uBegin; thetaBegin], [rFinal; vFinal;
uFinal; thetaFinal]];
problem.guess.control = [0, 0];

% Select a solver:
problem.options(1).method = 'trapezoid';
problem.options(1).trapezoid.nGrid = 8;
problem.options(1).MaxIterations = 1000;
problem.options(2).method = 'trapezoid';
problem.options(2).trapezoid.nGrid = 16;
problem.options(2).MaxIterations = 1000;
problem.options(3).method = 'hermiteSimpson';
problem.options(3).hermiteSimpson.nSegment = 15;
problem.options(3).MaxIterations = 1000;


% Solve the problem
soln = optimTraj(problem);
t = soln(end).grid.time;
r = soln(end).grid.state(1,:);
v = soln(end).grid.state(2,:);
u = soln(end).grid.state(3,:);
theta = soln(end).grid.state(4,:)*rtd;
gamma = soln(end).grid.control*rtd;

%plot on the heliocentric plot
figure(3);
hold on;
plot(r.*cosd(theta),r.*sind(theta),'k--');
%legend('Outbound Path of craft','Path of Mars','Path of Earth','Path of
Venus','Sun','Position of Earth at Earth Departure','Position of Mars at
Earth Departure','Position of Earth at Mars Arrival','Position of Mars at
Mars Arrival','Position of Earth at Mars Departure','Position of Mars at Mars
```

```
Departure','Position of Earth at Earth Arrival','Position of Mars at Earth
Arrival','Trajectory of Craft Directly after Mars Departure','Impulsive
Lambert Trajectory','Inbound Path of Craft','location','bestoutside');
legend('Outbound Path of craft','Path of Mars','Path of Earth','Path of
Venus','Sun','Position of Earth at Earth Departure','Position of Mars at
Earth Departure','Position of Earth at Mars Arrival','Position of Mars at
Mars Arrival','Position of Earth at Mars Departure','Position of Mars at Mars
Departure','Position of Earth at Earth Arrival','Position of Mars at Earth
Arrival','Inbound Path of Craft','location','bestoutside');

% Plot the solution:
figure(5); clf;

t = t*tcf;

subplot(5,1,1)
plot(t,gamma,'b-*')
ylabel('\gamma (\circ)')
title('Trajectory Parameters of the Inbound Path');
grid on;

subplot(5,1,2)
plot(t,r,'b-*')
grid on;
ylabel('r (AU)')

subplot(5,1,3)
plot(t,29.79*v,'b-*')
grid on;
ylabel('V (km/s)')

subplot(5,1,4)
plot(t,29.79*u,'b-*')
grid on;
ylabel('U (km/s)')

subplot(5,1,5)
plot(t,theta,'b-*')
grid on;
ylabel('\theta (\circ)')

bntep_mass(5) = bntep_mass(4) - max(t)*mdot;
bntep_delta_v(4) = (9.81/1000)*5000*log(bntep_mass(4)/bntep_mass(5));

fprintf('\nThe initial mass of the craft in Low Earth Orbit is %.3f
kg.\n',bntep_mass(1));
fprintf('The mass of the craft after it departs Earth is %.3f
kg.\n',bntep_mass(2));
fprintf('The mass of the craft as it arrives at Mars is %.3f
kg.\n',bntep_mass(3));
fprintf('The mass of the craft after it enters and leaves Mars is %.3f
kg.\n',bntep_mass(4));
fprintf('The final mass of the craft as it arrives at Earth is %.3f
kg.\n',bntep_mass(5));

burn_rate = 110000*3*(1/(950*9.81)); %kg/s for three NERVA type engines

mars_orbital_injection_dv = bntep_delta_v(3)/2;
```

```
mars_orbital_mass =
bntep_mass(3)/(exp(mars_orbital_injection_dv*1000*(1/(9.81*950))));

bntep_burn_time(1) = ((bntep_mass(1) - bntep_mass(2))/burn_rate)/3600; %hours
bntep_burn_time(2) = (((bntep_mass(3) - mars_orbital_mass))/burn_rate)/3600;
%hours
bntep_burn_time(3) = ((mars_orbital_mass - bntep_mass(4))/burn_rate)/3600;
%hours

fprintf('\nThe initial burn from LEO will require the NTP system to burn for
%.2f hours.\n',bntep_burn_time(1));
fprintf('The burn for Mars orbital injection will require the NTP system to
burn for %.2f hours.\n',bntep_burn_time(2));
fprintf('The burn to depart from Mars SOI will require the NTP system to burn
for %.2f hours.\n',bntep_burn_time(3));
fprintf('The total burn time for this mission %.2f
hours.\n',sum(bntep_burn_time));

fprintf('\nThe delta-v required to leave LEO is %.3f
km/s.\n',bntep_delta_v(1));
fprintf('The delta-v required to reach Mars using the NEP system is %.3f
km/s.\n',bntep_delta_v(2));
fprintf('The delta-v required to orbit and subsequently leave Mars is %.3f
km/s.\n',bntep_delta_v(3));
fprintf('The delta-v required to reach Earth using the NEP system is %.3f
km/s.\n',bntep_delta_v(4));
fprintf('By comparison, the delta-v required by an impulsive burn at Mars is
%.4f km/s.\n', norm(dvi));
fprintf('The total delta-v required for the mission is %.3f
km/s.\n',sum(bntep_delta_v));

fprintf('\nThe total duration of this mission is %.2f days.\n',outbound_time
+ stay_time + max(t));
```